OPTIMAL DESIGN OF GENERALIZED MULTIPLE MODEL ADAPTIVE CONTROLLERS

DISSERTATION

Thomas E. Brehm, B.S.E.E., M.S.C.E.G.

AFIT/DS/ENG/04-01

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/DS/ENG/04-01

OPTIMAL DESIGN OF GENERALIZED MULTIPLE MODEL ADAPTIVE CONTROLLERS

DISSERTATION

Presented to the Faculty

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

Thomas E. Brehm, B.S.E.E., M.S.

March 2004

AFIT/DS/ENG/04-01

Optimal Design of Generalized Multiple Model Adaptive Controllers

Thomas E. Brehm, B.S.E.E., M.S.

Approved:

/Signed/

Peter S. Maybeck
Committee Chairman

Date

/Signed/

Sharon A. Heise
Committee Member

Date

/Signed/

Meir Pachter
Committee Member

Date

/Signed/

Glen P. Perram
Dean's Representative

Date

Accepted:

/Signed/

Robert A. Calico, Jr.
Dean, School of Engineering and Management

Abstract

Advanced analysis and optimal design techniques that achieve performance improvement for multiple model adaptive control (MMAC) and multiple model adaptive estimation (MMAE) based control are developed and tested for this dissertation research. An adjunct area of research yielded modified linear quadratic Gaussian (LQG) control design techniques that also can be applied to nonadaptive control.

For the Modified LQG (MLQG) controller, the proposed designs remove the assumption that the Kalman filter as the observer and the controller gain matrix design are necessarily based on the same model as the best system model. The filter and controller gain matrices are both determined by models possibly other than the system model. In order to achieve optimal performance, the interrelationship of the system model to the filter and controller design models is established by minimizing a position correlation (mean square error on output) measure. Enhanced robustness is realized by considering the performance over the range of values of specified parameter(s) of the system model.

The proposed modified MMAC (M^3AC) architectures use the MLQG controller as the elemental controller in the MMAC. The performance improvements for the MLQG controller carry over to the M^3AC architectures as well as to the MMAE-based control architecture. Further study has established that the MMAC is essentially a special case of MMAE-based control. Both architectures are identical *in form* at steady state, which is critical to their design. Design approaches developed for the M^3AC are applied to the MMAE-based control with similar performance improvements.

Optimal design and analysis techniques for the MMAC and MMAE-based control resulting from this research are applied to a two-state system in which a single parameter is variable over a specified range of values. Though simple in nature, the two-state problem is representative of real-world applications. Analyses of the new design implementations demonstrate the performance improvements of the proposed architectures by comparing the results with those of the typical MMAC and LQG implementations.

Though incidental to this research, the performance enhancement of the MLQG controller itself has proven to be significant. The possibilities for application to non-adaptive control transcend this research into multiple model adaptive control. However, the techniques of the MLQG applied to the elemental controller in the MMAC and the analogous MMAE-based control result in considerable performance improvements as well.

AFIT/DS/ENG/04-01

To My Wife and Children

Acknowledgments

In this academic pursuit, it has not been so much as getting to the destination as much as it has been the journey. Looking back, it has been rewarding, fulfilling and fun. During this time, I have always tried to maintain a balance between that which is important to me; family, work, academics, and the Lord. Thus, I have many to thank for helping to this destination.

First, I would like to thank my dissertation committee, Meir Pachter, Sharon Heise and the dean's representative, Glen Perram. I especially would like to thank my dissertation advisor, Peter Maybeck for his guidance, advice, technical input and probably most important, his patience. It is clear that his faith in the Lord guides even his professional endeavors. I have not only learned much, I am truly a better person for working with him.

I would like to thank my coworkers at the National Air Intelligence Center who have offered support and advice in this endeavor. Also, I thank the leadership who gave me the opportunity through the long term full time training program to finish the course work.

I am truly indebted to my parents who have given me so much that I will never be able to thank them enough. Their dedication to family, God, and hard work continues to be an inspiration to me and serves as a model to conduct my life.

Time spent on this work has been some time spent away from my family. To my wife, I love you and I so very much appreciate the sacrifices you made to make this possible. You have been there always for me, even when I have not in kind because of this work. I thank God for you. To my children, precious gifts from the Lord, I can only hope that someday you will understand the sacrifices you have made. I thank you for helping me to always keep things in perspective.

Finally, I thank the Lord, who has during this journey, carried me, placed people to guide me, and has grown my faith in Him. "I have the strength for everything through Him who empowers me" - Philippians 4:13.

Table of Contents

List of Figures

List of Tables

Notation

Scalars, Vectors, Matrices

Scalars are denoted by upper or lower case letters in plain type, as the scalars x or J.

Vectors are denoted by lower case letters in boldface type, as the vector **x** made up of components $\mathbf{x}_i$.

Matrices are denoted by upper case letters in boldface type, as the matrix **A** made up of elements $\mathbf{A}_{ij}$ (*i*th row, *j*th column).

Subscripts

$_f$:	filter model
$_k$:	the output of the *k*th filter model
$_t$:	truth model

Superscripts

^ :	estimated, computed, or measured value (not true value)
$^{-1}$:	matrix inverse
^{T} :	vector or matrix transpose

Acronym List

Abbreviation	**Acronym**
AFIT	Air Force Institute of Technology
GMMAC	Generalized Multiple Model Adaptive Control
IRDF	Inter-Residual Distance Feedback
LQG	Linear Quadratic Gaussian
MAP	Maximum a posteriori
MLQG	Modified Linear Quadratic Gaussian
MMAC	Multiple Model Adaptive Control
MMAE	Multiple Model Adaptive Estimation
M^3AE	Modified Multiple Model Adaptive Estimation
RMS	Root-Mean-Square
USAF	United States Air Force

Optimal Design of Generalized Multiple Model Adaptive Controllers

Chapter 1 - Introduction

1.1 Overview

Adaptability and robustness are two of the desirable attributes associated with synthesizing an advanced controller. Each has been the focus of extensive and broad range of research and each has its own merits for consideration. Multiple Model Adaptive Control (MMAC) and Multiple Model Adaptive Estimation (MMAE) based control, both originating from stochastic control, are two similar approaches to adaptive control that have demonstrated success, and the performance improvement of each is the subject of this dissertation. MMAC adapts to specific parameters by using a bank of LQG controllers designed for a predetermined set of models, i.e. for a set of discrete parameter values. Residual information from the Kalman filters is used to compute a probability-based weighting on the controllers that yields an overall control command that is best matched for the current value of the uncertain parameters. MMAE-based control uses a bank of Kalman filters that can provide both state estimates and parameter estimates. The state estimates are fed to a full-state feedback control gain matrix to yield the control signal. The full-state feedback gain matrix itself is determined based upon the parameter estimate. Until this point, robustness enhancement of the MMAC and MMAE-based control architectures has not specifically been addressed and is an additional subject of this research.

The Kalman filter and controller components of the MMAC and MMAE-based control are based on typical LQG synthesis methods. These techniques rely on the separation principal that assumes the models for design of the filter and the controller are equivalent to the system model and that each component is designed separately. This research has produced performance improvements of the MMAC and MMAE-based control architectures by applying newly developed techniques for LQG design. This new design approach is based on using models for implementation of the Kalman filter and gain matrix of the full-state feedback controller that are different from the system model. These enhancements do not involve modifications to the MMAC or MMAE-based control architectures, but only the change in component design. Further research discussed in this dissertation involves the modification of both architectures as well, in order to improve performance.

The second issue addressed in this dissertation is robustness of the MMAC and MMAE-based controllers. Robustness can be categorized as either robustness of the adaptation, or robustness to those parameters to which the controller does not adapt. This research investigates the former. For robustness of an adaptive architecture, one could argue that, if it adapts perfectly to the specific uncertain parameters, then it is robust to any variation of those parameters. For both multiple model control architectures, this is possible only if there is a model that matches the value of the uncertain parameter(s). Obviously, over an uncertain parameter space, there would have to be an infinite number of models to match any variation exactly. Only a small, finite number of models is computationally possible. This research demonstrates that within the MMAC and MMAE-based control architectures it is indeed possible to incorporate robustness to the adapted parameters with the constrained number of models for the uncertain parameter space.

Finally, this research is being conducted without a specific target application, but with a simple two-state problem upon which to verify theoretical results. The two-state problem, much smaller than typical applications, should facilitate analysis of the results.

This chapter continues with a more in-depth discussion of the MMAC, MMAE-based control and related adaptive control architectures currently being researched and implemented. Next, the approach to this research and contributions to the field are discussed. Finally, the last section outlines the rest of the dissertation.

1.2 Background

Adaptive control is an area that has generated an extensive and widely varied number of techniques. One way of classifying adaptation approaches is by the amount of assumed information. A system identification approach might be as general as determining the order of the plant, fixing a model, and then identifying the unknown parameters in the model. Of course, the performance of the implementation of such an approach will depend on the amount of data available and available computer time. Such an extensive and general approach may be theoretically intractable and impossible to accomplish in real time. At the other end of the spectrum are direct adaptation control techniques based on an assumed model or set of models with assumed but fixed parameters for the unknowns. Given some criterion, the models are then judged for their output's closeness to the current plant operating conditions. While some approaches will simply use the closest single model for the control, others may use a blending of the closest models since no one model may be an exact match. Adaptive control is a wide area of research and this section covers some of the most prevalent and promising techniques. This section begins with a more in-depth description of MMAC and MMAE-based control, the control architectures used in this research.

1.2.1 Multiple Model Adaptive Control and MMAE-Based Control

The basic structure of an MMAC [29,35] is shown in Figure 1.1. There are three main components: multiple model estimators (MME), hypothesis conditional probability computation, and control computation. The MME is typically a bank of Kalman filters running in parallel in which each filter is designed for an assumed value of the parameters within the model. For each model, the MME outputs the state estimate and the residual associated with each model. The control computation uses the state

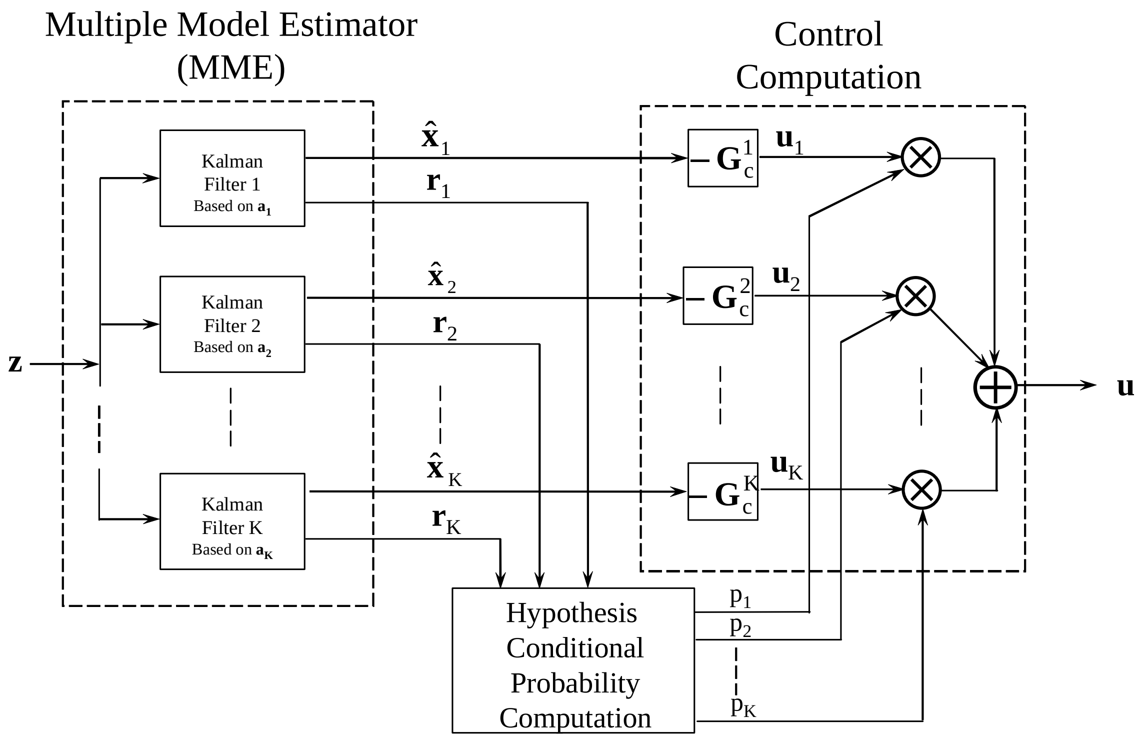

Figure 1.1 Generalized multiple model adaptive control structure

estimates, and the hypothesis conditional probability computation uses the residuals. The hypothesis conditional probability computation determines a weight based on the residuals of the parallel elemental filters. This weight calculation essentially corresponds to the apparent correctness for each model at the current time, based on the measurement history seen up to the current time. The control computation is a bank of LQ full-state feedback controllers in which each controller is designed for the corresponding model in the MME. Control output can be computed by one of two methods. The first method is the Bayesian form in which individual control components are blended by the probability weight computed in the previous block. The second method is the maximum a posteriori (MAP) form in which a single Kalman filter and associated LQ controller with the highest probability is selected.

The multiple number of filters running in parallel gives the MMAC adaptation speed, but also forces design issues. The Kalman filters can be computationally intensive, which limits the practical number of models. Thus, a small number of filters can be used (unless truly parallel computation is available, so that any number of parallel elements can be used with no impact on computation time). The design issue becomes designation of the assumed parameter values, i.e., where the discrete points (used for the basis of defining filters) should be placed in the parameter space. Sheldon [56,57] solved this problem with his work. As will be discussed, it has also been shown [56,57] that optimal discrete parameter point placement is also dependent on the goal: whether parameter estimation, state estimation or control. For the architecture shown in Figure 1.1, state estimation, parameter estimation, and control output are easily obtained, but the end goal for MMAC is best control. In fact, the MMAC never explicitly forms the overall adaptive state estimate or parameter estimate.

The individual controllers that are blended with the probability weights are designed based on the assumed parameters for the underlying models. For the MMAC, LQG has been used exclusively for the controller synthesis and its viability has been demonstrated in many applications [1,16,17,19,20,39,52,53,58,59,60]. The design concept also assumes separation between the estimator and the full-state feedback control elements of the controller [36]. Invoking this principle allows these two components of the controller's structure to be designed independently. Sheldon's approach [56] does consider the full-state feedback controllers and estimators in separate steps when optimally placing the models (i.e. placing the discrete points in parameter space).

State and parameter estimation can be accomplished using multiple model adaptive estimation (MMAE) techniques, which can be used for another control technique known as MMAE-based control [62], shown in Figure 1.2. The state estimate is derived from the individual state estimates output by the bank of Kalman filters. The overall adaptive state estimate is determined with a probability-weighting equivalent to that used in the MMAC's control computation. Parameter estimates are calculated using the same probability weighting on the assumed model parameters. An MMAE-based control approach uses these parameter estimates to identify a full-state feedback controller gain to be used for the system; multiplying the negative of that gain by the MMAE state estimate generates the control. Again, forced separation is assumed which allows independent implementation of the estimator and full-state feedback controller. Sheldon's approach can identify how to place the models (discrete parameter values) for

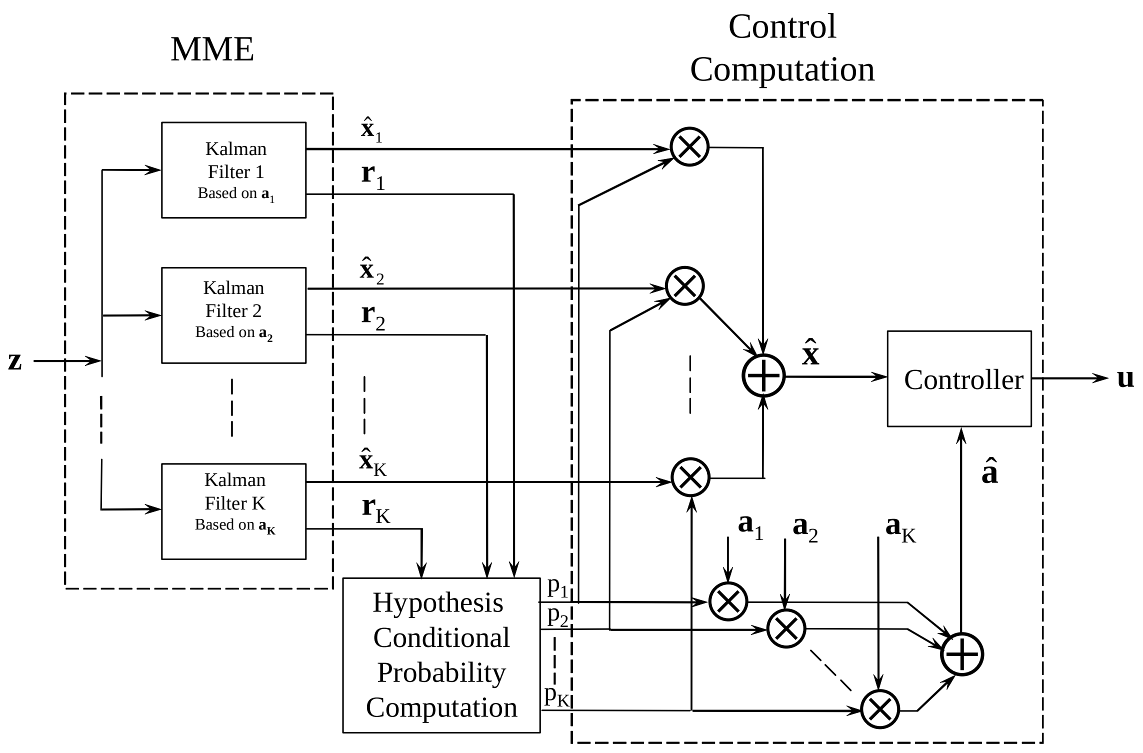

Figure 1.2 Multiple model adaptive estimation based control architecture

best parameter estimation or for best state estimation, but not both simultaneously [56]. The question remains, how to specify the independently designed controllers. Should the controller be designed on-line or should there be a look-up table of fixed controllers? This question is similar for estimation, and that was answered by Miller [44]: a single Kalman filter outside the MMAE structure, based upon the parameter estimate from the MMAE. However, in that case, a Kalman filter was designed on-line, whereas for control, the answer may not be similar due to backward solutions of the Riccati difference equations which are required for control synthesis.

Work by Vasquez [63,64,65] used a Sheldon-like algorithm repeatedly in real time to place the models to enhance the estimation. This is one form of the moving-bank MMAE [37]. For the usual type of moving-bank approach, there is a fixed *number* of filters in the bank that are designed based on specified parameters, and which particular parameter values can be dynamically redeclared in real time. As the system changes, the initial bank may not represent the optimal placement of filters. The bank can expand, contract, or move in the parameter space in order to achieve a "better" (but not necessarily optimal) model placement. Vasquez has improved the algorithm by not only enhancing the move logic, but also by applying the Sheldon-like algorithm in real time so that the model placement is optimal. This algorithm could be used in the MMAE-based control scheme to obtain the best parameter estimate in order to select or design the applicable controller better than in other current moving-bank controller approaches.

1.2.2 Related Work in Multiple Model Control

The two approaches to adaptive control to be discussed in this section are based on assumed knowledge of the plant to control, described in terms of models, rather than based on system identification techniques to identify the system model. These techniques are model reference adaptive control (MRAC) and multiple model switching control (MMSC). They are discussed with respect to how they relate to the MMAC and MMAE-based control approaches. MRAC is a method that essentially assumes a model of the controller as a reference and uses a form of system identification to make changes to the controller according to deviations from the assumed parameters. MMSC has a set of plant models from which is selected the single one for the basis of the online controller.

Given a system in which parameters vary or are unknown, the objective of model reference adaptive control (MRAC) [10,14,25,43,61] is to obtain a system response based on a given model of the desired performance of the system. For this scenario, the input and the reference model are known and the output response is measured. Thus, the exact characteristics of the system do not have to be known at any instant. It is only required to have the closed-loop performance characteristics match those specified in the reference model. Robotics is an excellent and common application of MRAC [10,25,62] since the loads for such systems are usually variable. Even if the load is constant, due to the nature of the physical structure, the dynamics that describe the system will change during operation. The operating point is not constant, but may vary slowly enough in comparison to the desired response, that the system could adapt.

Most notably missing from the basic algorithm is the handling of measurement noise. In the robotics example, this exclusion may be allowable for position sensing, but velocity sensing can be very noisy. Some recent research has developed techniques to address measurement noise [14,43,51]. Another area for research is the robustness of the identification algorithm [61]. The identification algorithm is at the heart of MRAC and accordingly, the focal area of research.

The advantage to this approach is that the exact values of the system parameters do not have to be known. In addition, since the control algorithm is adaptive, the system can vary, though slowly relative to the adaptation algorithm and the desired system response. However, this is also the disadvantage of MRAC. For a rapidly varying system, the adaptation mechanism may not be fast enough. The implementation of the algorithm can be computationally intensive, which can also be a problem. Faster and robust algorithms are necessary. Robustness could alleviate the requirement for fast adaptation.

The MRAC approach essentially uses one model and the algorithm computes a control to compensate for deviations, rather than selecting a model that has a controller that produces the desired response, as is the case with MMAC. An area of current research is concerned with computational approaches to compensation for deviations from the assumed model. For the MRAC, since the adaptation algorithm adjusts for deviations from the assumed model where the resultant gain adjustments are continuous, it could be considered that the control is selected from an infinite number of models.

Another model-based approach to adaptive control is multiple model switching control (MMSC) [8,9,46,47,48,49,50]. For this architecture, the goal is to provide rapid adaptation to parameter uncertainties in a given plant. However, unlike the model reference control in the previous section, the approach of MMSC is to use more than one model to represent the plant in a closed loop system. It is assumed that one of the models in the structure along with its corresponding controller will provide the desired response. This requires coverage of the uncertain parameter space by a sufficient population of models. The dispersion of the models and the algorithms for switching between the models in the uncertain parameter space is the focus of current research [8,9,46,47,48]. As will be discussed in the sequel, this is directly related to the MAP approach to MMAC.

The general structure has three major component blocks: the multiple models, identifiers, and controllers. The multiple models of the plant are at the center of the structure with a controller designed for each of the multiple models. The identifier outputs for the models are compared to the actual output of the plant. This error difference is used in the switching portion of the algorithm to choose a single controller to generate the control action. The controllers are designed to correspond to the specific models. Though all controllers operate in parallel and all model information is available, only one model or hypothesis affects the actual control applied to the plant at any one instant. The controller used will only change if the error grows such that another model becomes a better representation of the current operating point.

The basic approach assumes all models are fixed and they do not adapt to parameter change. Variations to this structure allow for a combination of fixed and adaptive models [48]. Thus, three combinations are possible: all fixed models (as discussed), N adaptive models and M fixed models, and all adaptive models. Clearly, since there is a finite number of models, it is unlikely that any single model will exactly duplicate the current plant. In order to overcome model mismatches, one of the models may be adaptive such that its finely tuned parameters provide better performance. As parameters slowly vary, the adaptive model will follow those changes. When there are large parameter changes, the adaptive model's initial value will be that of the newly switched model. The aforementioned combinations, except for all adaptive models, have been proven to be stable [48].

This approach has been demonstrated with an application to robotics [8,9,47]. The three combinations of fixed and adaptive models were tested. It was found that, if the transient response could be improved by a rapidly adaptive model, the switching environment could be faster. This in turn means that the actual system can change parameter values more rapidly. This has the same drawbacks as the MRAC scheme because the adaptive portion is dependent on the efficiency of the algorithm and computation speed. The adaptive portion has to be faster than the switching or the response will never reach steady state in order to perform switching.

MMSC, when used with fixed models, has the advantage that only the response of the individual models has to be calculated and not an on-line optimization, as is the case of MRAC. However, if the actual system does not correspond to an existing model, the transient response will be poor. Hence there is a need to add an adaptive portion. MMAC on the other hand has a blending of the control of the closest models, thus providing some approximation to the actual system parameters when the parameters are not consistent with the fixed models. In addition, placement of the models (i.e. placement of the discrete points in parameter space to be used for the basis of the models) is a major concern in the MMAC research [57]. This does not seem to be the case with MMSC, whereas it seems that it should be. Perhaps the inclusion of an adaptive portion reduces the emphasis on model placement. Additionally, the reader should conclude that the MMAC and MMSC are very closely related.

The MMAC approach has the benefit of quick adaptation to compensate for variations in specific parameters within the system model. It is assumed that these uncertain parameters will vary over a given range. For practical problems, there is some knowledge of the plant and its possible variations in parameter values. This has been demonstrated in many successful applications of the MMAC [1,16,17,19,20,39,52,53,58, 59,60,] and there has been much research in refining the theory [37,56,57]. Not only are multiple model techniques used for control, a similar approach can be applied to state and parameter estimation, as has been demonstrated in the research and application of multiple model adaptive estimation (MMAE) [7,11,12,21,24,26,27,38,41,42]. In fact, as discussed, MMAE-based control is a blending of MMAE state and parameter estimation with full-state feedback control (based on estimated parameters) in order to yield an MMAC-like structure.

As compared to the previous two approaches, adaptation is simply the blending of the control output based on the correctness of the possible models. Since the Kalman filters run in parallel, all information for the blending is immediately available for the control calculation through the residuals of the filters. The MRAC scheme involves constant identification of the system parameters in order to adjust the controller. The key is adaptation through reoptimization of the single controller. For the MMAC, it is emphasized that the *individual* controllers do not adapt, but the filters provide constant information for adaptation through blending of the fixed controllers. The MMSC approach is very similar to the MMAC approach. In fact, the switching control approach can be closely duplicated by the MMAC if only the controller with the highest probability is used. This MMAC variation is referred to as the MAP approach [35] to controller selection, as will be discussed subsequently.

Finally, despite claims from researchers in MMSC, there are stability proofs available for the MMAC approach [15]. Proof of stability does not guarantee anything about performance. Sheldon's optimization of model placement will place the models to yield the best performance with respect to one of the three criteria, RMS state estimation errors, RMS parameter estimation errors, or RMS control regulation errors.

1.3 Research Objectives and Contributions

The objective of this research is to demonstrate performance improvements and robustness enhancement in adaptation of MMAC and MMAE-based control structures through modifications of current design approaches and typical architectures. As stated, synthesis of multiple model control structures uses LQG techniques for design of the elemental components of the architecture. The first area of research removes the assumption, inherent in the LQG synthesis, that the design model for the Kalman filter portion is the same as the design model for the full-state feedback controller portion. Further, either of these two models may be different from the truth model of the real-world system. The second research area removes the assumption that LQG synthesis is used for the design of the individual components of the multiple model control structures. Lifting this assumption opens many possibilities to investigate for the basic filter and control elements and their interdependencies in the multiple model structures. Finally, the synthesis approaches developed in the previous two areas of research require a method to determine where the discrete points (used for the basis of defining filters *and* controllers) should be placed in the parameter space. This development is an extension to the approach developed by Sheldon [56,57] in his work.

1.3.1 LQG Design Contributions

Inherent in the synthesis of the LQG controller is certainty equivalence, or the separation principle, which stipulates that though the Kalman filter and the full-state feedback controller modules are designed separately, the design models are the same as the truth model for the real-world system. Removing the assumption that the filter design model is the same as the truth model leads to an investigation into a modified LQG controller based upon the following hypothesis:

> *Hypothesis 1: An LQG controller that has the best performance for minimized regulation error may have a filter that is designed based on a model (parameter value) that is not necessarily the same as the model (parameter value) for the full-state feedback controller synthesis or the "best" model of the real-world system.*

To test this hypothesis, the approach is to develop a performance measure that incorporates the filter and full-state feedback controller into a single evaluation. This performance measure is then used in an optimization algorithm that will select the best filter, given an LQ full-state regulator controller designed for the specified truth model of the system. From this work comes the following contribution:

> *Contribution 1: A design algorithm that yields a modified LQG that minimizes regulation error and at minimum, performs as well as the typical LQG control for the same criterion.*

Inherent in the first hypothesis is that the *best* real-world system model is known and it is assumed that it deviates minimally from the nominal truth model. Further, it was assumed that the model for the full-state feedback controller synthesis was the same as the truth model. Under these assumptions, robustness to deviations of specific parameters in a system is not explicitly considered in the design algorithm. Any differences between the assumed truth model and the actual system model will affect performance. The approach to address the previous hypothesis was that the filter model might be different from the system model. Now consider that *both* the filter and full-state feedback controller design models may be different from the nominal truth model of the real-world system. Also, consider that specific parameters of the system model may deviate over a given range. These considerations lead to the following hypothesis:

> *Hypothesis 2: A modified LQG controller that is robust to deviations to a nominal system model may have a filter designed for a model that is different from the model for the full-state controller synthesis. These two design models <u>both</u> may be different from the nominal truth model of the real-world system.*

The work to validate this second hypothesis follows directly from the first. The performance measure does not change since the original performance measure includes the filter, the controller and system models. For a modified LQG controller with robustness, the optimization algorithm changes. The criterion for optimality changes from being based on a single evaluation of the performance measure for a truth model to an evaluation of performance over a (bounded) range of possible truth models. This bounded range of truth models contains the possible truth model of the real world system at any point in time. The model for the LQ synthesis of the controller may be one of these possible truth models, but not necessarily the same as the nominal truth model. This algorithm development yields the second contribution:

Contribution 2: A design algorithm that yields a modified LQG with robustness to deviations to the nominal model of the real-world system that minimizes regulation error. At a minimum, the modified LQG with robustness performs as well as the typical nonadaptive LQG control for the same criterion.

1.3.2 MMAC Analysis and Design Contributions

Since the fundamental elements of the MMAC are based on the LQG controller, the work from the first and second contributions leads to development of several modifications to the typical MMAC architecture. Consider the modified LQG controller from the first contribution. If the performance of an LQG controller can be enhanced, then it seems reasonable that replacing the elemental controllers in the MMAC with the modified LQG controllers should enhance the performance of the MMAC. Replacing the typical LQG controllers in the MMAC with the modified LQG controllers is expressed in the following hypothesis:

Hypothesis 3: For an MMAC that has the best performance for minimized regulation error, the filters in the MMAE substructure may have design models that are different from their corresponding full-state feedback control gain matrices.

In this architecture modification, the filters in the MME portion still provide the necessary residual information for the probability weighting on the control output of the individual full-state feedback controller elements. However, the design models on which these filters are based are not necessarily the same as the design models for the full-state feedback gain matrices. The optimal placement of the filter model parameter values (discretization) does not change from the original MMAC discretization algorithm except to incorporate the procedures of the modified LQG controller. This work leads to the following contribution:

Contribution 3: A design algorithm that yields a modified MMAC that minimizes regulation error and at minimum, performs as well as the typical MMAC.

Now consider the modified LQG controller with robustness from the second contribution to enhance robustness of the MMAC. One aspect of the MMAC that will be discussed in more detail in the following chapters is that at steady-state, probability will flow to essentially one filter. Thus, under this condition, essentially one LQG control element will be effective at steady-state and it is not guaranteed to match the actual system model. Another possible condition is that any deviations of the real world system from the assumed truth model might not trigger changes in the probability weighting once the MMAC has reached steady state. Under these two conditions, if the MMAC reaches steady-state and does not further adapt, then performance is related directly to the robustness of the individual LQG controller elements. Thus, it is possible that the modified LQG controller with robustness can improve the performance of the MMAC architecture. This is expressed in the following hypothesis:

> *Hypothesis 4: An MMAC that uses modified LQG-with-robustness controller elements will be robust to variations in the assumed system model to which the MMAC does not adapt and mismatches between the assumed system model and elemental LQG controller model selected at steady state.*

To validate this hypothesis, the LQG elemental controllers in the MMAC are replaced with the modified LQG controllers with robustness from Contribution 2. The architecture design requires a modification of the discretization algorithm to account for the robust LQG elements and the possible differences between assumed system model and the filter models. Development of this new architecture yields the following contribution:

> *Contribution 4: A design procedure that yields a modified MMAC with robustness to differences between the nominal system model and the steady state filter model in the MME portion of the MMAC.*

The previous two contributions do not modify the MMAC architecture, just the design of the individual components. The previous discussions assume that each elemental controller in the MMAC is a full-state feedback gain matrix tied directly to a single Kalman filter in the MME. The residuals from the filters are used to determine the probability weighting for blending of the elemental controllers' output. Thus, the Kalman filters are used to compute the control as well as *decide* the control to apply. Implicit in discretization (model parameter placement) is a trade-off between controller selection and control computation. A proposed solution to this trade-off is to lift the assumption that the control is determined by a simple gain matrix multiplication and replace it with an LQG controller that receives measurement data. This step allows the LQG controllers to be designed separately from the MME portion of the MMAC. Still, the control output from the elemental LQG controllers are blended according to the probability weighting computed from the MME residual information. Admittedly, this architecture change makes the MMAC look like a MMAE-based controller, however, the control is blended after elemental control computation versus blending states and then computing control. This proposed architecture modification is stated in the following hypothesis:

Hypothesis 5: In order to obtain complete separation between the filter elements in the MME and full-state feedback control elements of the MMAC, any state estimation in the control element must be independent from the filter in the MME. This structure is a complete generalization of the MMAC and the typical MMAC can be obtained as a special case.

The structure that is implied in this hypothesis is more complex since it maintains a second set of Kalman filters. The Kalman filters for the control portion do not have to maintain all the information to compute the probability weights. Thus, the complexity is not necessarily doubled. The performance measure that is used for the typical MMAC has to be augmented to include the LQG control elements. In addition, the algorithm to minimize the performance measure must be modified to allow for MME filter design models to be different from the LQG controller design models.

Contribution 5: A design procedure that yields a generalized MMAC that has LQG controllers for the control elements separate from the bank of Kalman filters in the MME portion.

1.3.3 MMAE-Based Control Analysis and Design Contributions

Three areas of research into MMAE-based control are addressed in this dissertation, each of which yields a contribution to multiple model control. The first area is analysis of the MMAE-based control in relation to MMAC. This analysis gives insight into choosing one implementation over the other. The second area of research is a method for discretization of the parameter space, similar to what is already available for the MMAC. When MMAE-based control (or any other multiple model architecture for that matter) is chosen for implementation, a discretization method is necessary to obtain the best design. Finally, an architecture change to the typical MMAE-based control is proposed. This architecture change overcomes certain implementation trade-offs to be discussed for the typical MMAE-based control.

From the previous discussions, it should be evident that the MMAC and MMAE-based control architectures are closely related. Again, the basic difference between the two architectures is that, for MMAC, control is computed by elemental controllers and then blended using probability weighting, while for MMAE-based control, the state estimates are blended and then control is computed. Since the MMAE-based control and MMAC architectures are similar, analysis reveals that there are certain conditions under which they will perform similarly and also how they perform differently. This analysis will give insight to the conditions under which one approach is preferable to the other. These findings come from the work that is motivated by from the following hypothesis:

Hypothesis 6: Under certain assumptions and conditions, the MMAE-based control architecture performs essentially the same as the MMAC architecture. Identifying these conditions will aid in making engineering decisions for implementation.

To validate this hypothesis, first, analytical expressions for both MMAC and MMAE-based control are derived. Next, the expressions are analyzed to determine where approximations can be made or conditions imposed for making the expressions produce similar results. Further analysis of these conditions reveals when one approach may produce better results or may be computationally advantageous. This work yields the following:

> *Contribution 6: A framework is developed for analyzing MMAC and MMAE-based control in order to make engineering decisions to determine which architecture to implement.*

Inherent in the MMAE-based control are design trade-offs that are in part due to the implementation of the MMAE portion. As will be discussed in subsequent chapters, the MMAE can only be optimized to yield the best parameter estimate or the best state estimate, but not both. The parameter discretization for the MMAC is based on an optimization of a control criterion. A similar discretization method for the MMAE-based control, based on a control criterion, is proposed in the following hypothesis:

> *Hypothesis 7: There exists a parameter discretization method for the MMAE-based control architecture that optimally places the models in the MMAE portion for best control performance.*

To validate this hypothesis, Sheldon's approach for discretization of the parameter space for the MMAC was modified. Given that a closed loop expression for the MMAE-based control performance can be developed, as was necessary for the last contribution, then it is possible to optimize the placement of models in the MMAE according to a control criterion. This work yields the following contribution:

Contribution 7: A discretization method for MMAE-based control that yields an optimal placement of the models in the MMAE with respect to a control performance criterion.

A second approach to eliminate the inherent problem of the MMAE optimization is to modify the architecture. It has been established that the MMAE portion can be optimized for best parameter estimation. Thus, it seems appropriate to use the parameter estimate to design and implement the best LQG controller online. This approach is summarized in the following hypothesis:

Hypothesis 8: The MMAE portion of the MMAE-based controller can be discretized for best parameter estimation. The parameter estimate can be used for an online implementation of an LQG controller. This architecture, at a minimum, will perform as well as the MMAE-based controller optimized for control.

To validate this hypothesis, the typical MMAE-based architecture shown in Figure 1.2 was modified such that the full-state feedback control gain fed by the blended state estimate is replaced with an LQG controller block fed by the same measurements as the MMAE portion. These changes are shown in Figure 1.3. The online design of the LQG controller and its performance in the overall architecture is the focus of this phase of the MMAE-based control research. This design approach is the MMAC analog of the M^{3}AE architecture proposed by Miller [44]. The MMAE structure is used to get the estimate $\hat{\mathbf{a}}$ which is used to design or select a single (generalized) LQG controller instead of a single Kalman filter, as in the M^{3}AE. Apparent from the previous contributions yielding the modified MMAC, the online design is not necessarily an LQG controller designed by the conventional method. This architecture is truly a separation of the

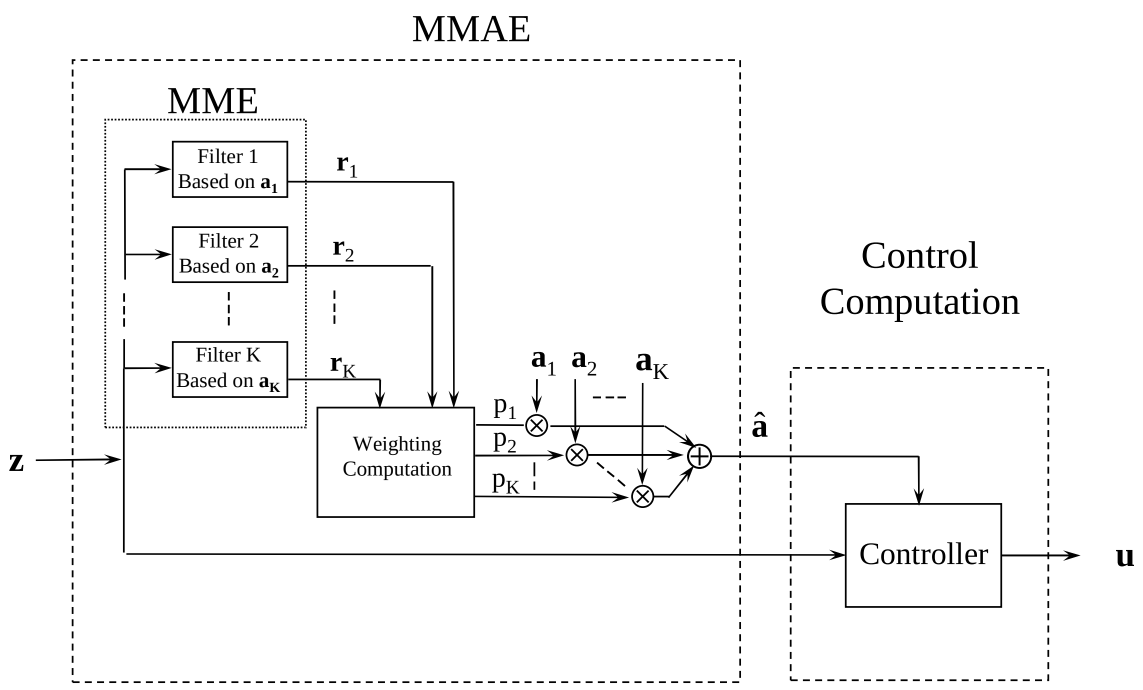

Figure 1.3 Modified MMAE-based architecture

controller selection from the control computation unlike the typical MMAE-based control. The results yield the following contribution:

> *Contribution 8: A modified MMAE-based control architecture that performs at least as well as MMAE-based architecture and allows versatility in the control scheduling for possible values of uncertain parameters of the system as determined by parameter estimates.*

1.4 Summary

This introductory chapter has given some of the preliminary background for the ensuing discussion of MMAC and MMAE-based control research. The stepwise approach for research has been outlined by a discussion of hypotheses and corresponding contributions. The subsequent chapters elaborate the results. First, more detail of MMAC and MMAE-based control is discussed in Chapter 2 to lay a more extensive groundwork for subsequent chapters. Chapter 3 discusses enhancements to the LQG design approach. Next, Chapters 4 and 5 discuss the enhancements to MMAC and MMAE-based control, respectively, as previously proposed. In Chapter 6, a simple two-state system is used to demonstrate the contributions of this research to a specific, insight-providing application. Finally, Chapter 7 summarizes the material in this dissertation and proposed further potential related research areas.

Chapter 2 - Multiple Model Adaptive Control Preliminaries

The multiple model approach to stochastic estimation was first proposed by Magill [29]. Not only can the states be estimated by an MMAE, but the uncertain parameters of the system can be estimated as well. The development of MMAC as a multiple model approach to stochastic control naturally follows, and so the discussion of MMAC builds upon the MMAE. A logical variation to MMAC is the MMAE-based control in which the estimated plant parameters are used to determine a single desired controller on-line, and then this controller operates on the MMAE-generated state estimates. There are variations to multiple model control in order to make implementation more practical. Finally, regardless of whether the purpose of the multiple model algorithm is estimation or control, the placement of the models in parameter space and the tuning of the individual elemental filters are vitally important and therefore, major research issues to be discussed in the sequel.

2.1 Kalman Filtering Basic Development

The Kalman filter is the basic component of both the MMAC and MMAE development. This discussion of the Kalman filter follows [34]. The linear discrete-time Kalman filter with sampled data measurements will be the standard form used in this research. The underlying assumption for the system is that it can be described by a linear stochastic state model driven by white Gaussian noise, yielding Gauss-Markov state processes with Gaussian state (and noise) probability density functions.

A physical system is usually continuous in time, but for the purposes of computer implementation, the equivalent discrete-time model [34] will be used. The system and measurement equations are as follows:

$$\mathbf{x}(t_i) = \mathbf{\Phi}(t_i, t_{i-1})\mathbf{x}(t_{i-1}) + \mathbf{B}_\mathrm{d}(t_{i-1})\mathbf{u}(t_{i-1}) + \mathbf{G}_\mathrm{d}(t_{i-1})\mathbf{w}_\mathrm{d}(t_{i-1}) \tag{2.1}$$

$$\mathbf{z}(t_i) = \mathbf{H}(t_i)\mathbf{x}(t_i) + \mathbf{v}(t_i) \tag{2.2}$$

where

$\mathbf{x}$	=	n-dimensional system state vector
$\mathbf{\Phi}$	=	state transition matrix
$\mathbf{B}_\mathrm{d}$	=	discrete equivalent of the system control input matrix
$\mathbf{u}$	=	r-dimensional deterministic control input vector
$\mathbf{G}_\mathrm{d}$	=	discrete equivalent of the noise input matrix
$\mathbf{W}_\mathrm{d}$	=	s-dimensional discrete-time zero-mean white Gaussian dynamics driving noise vector with covariance $\mathbf{Q}_\mathrm{d}(t_i)$
$\mathbf{z}$	=	m-dimensional measurement vector
$\mathbf{H}$	=	system output matrix
$\mathbf{v}$	=	discrete-time zero-mean white Gaussian measurement noise vector with covariance $\mathbf{R}(t_i)$

Nonlinear system requires an extended Kalman filter, which will not be specifically discussed here. Equations for the extended Kalman filter can be found in [35]. It is assumed that the system can be described by a sampled-data representation and accordingly, the Kalman filter form is discrete-time. The equations for the Kalman filter are the basic representation as found in e.g. [34] and given as follows:

$$\hat{\mathbf{x}}(t_i^-) = \mathbf{\Phi}(t_i, t_{i-1})\hat{\mathbf{x}}(t_{i-1}^+) + \mathbf{B}_\mathrm{d}(t_{i-1})\mathbf{u}(t_{i-1}) \tag{2.3}$$

$$\mathbf{P}(t_i^-) = \mathbf{\Phi}(t_i, t_{i-1})\mathbf{P}(t_{i-1}^+)\mathbf{\Phi}^{\mathrm{T}}(t_i, t_{i-1}) + \mathbf{G}_{\mathrm{d}}(t_{i-1})\mathbf{Q}_{\mathrm{d}}(t_{i-1})\mathbf{G}_{\mathrm{d}}{}^{\mathrm{T}}(t_{i-1}) \tag{2.4}$$

These equations propagate the state and covariance from just after the previous measurement update time to just before the current measurement. Measurement updates to the filter are accomplished with the following equations:

$$\mathbf{A}(t_i) = \mathbf{H}(t_i)\mathbf{P}(t_i^-)\mathbf{H}^{\mathrm{T}}(t_i) + \mathbf{R}(t_i) \tag{2.5}$$

$$\mathbf{K}(t_i) = \mathbf{P}(t_i^-)\mathbf{H}^{\mathrm{T}}(t_i)\mathbf{A}^{-1}(t_i) \tag{2.6}$$

$$\hat{\mathbf{x}}(t_i^+) = \hat{\mathbf{x}}(t_i^-) + \mathbf{K}(t_i)[\mathbf{z}_i - \mathbf{H}(t_i)\hat{\mathbf{x}}(t_i^-)] \tag{2.7}$$

$$\mathbf{P}(t_i^+) = \mathbf{P}(t_i^-) - \mathbf{K}(t_i)\mathbf{H}(t_i)\mathbf{P}(t_i^-) \tag{2.8}$$

$$\mathbf{r}(t_i) = \mathbf{z}_i - \mathbf{H}(t_i)\hat{\mathbf{x}}(t_i^-) \tag{2.9}$$

where the values **K**, **A** and **r** are the filter gain, the filter-computed residual covariance, and filter residual, respectively. The measurement is taken at time t_i, and the measurement is incorporated to give the state estimate and residual covariance just after time t_i.

2.2 Multiple Model Adaptive Estimation Development

This section details the development of MMAE. Though the focus of the research is MMAC, this MMAE development lays the foundation for the following sections on control. For MMAE and MMAC development, the original assumptions of the system model remain in place. However, there are two additional assumptions that address the uncertain parameters specifications. They are:

- Uncertain parameters in the system are constrained to the parameters describing the matrices in the state dynamics model, measurement model, or the statistics of the measurement noises entering the system, i.e., the uncertain parameter vector **a** can affect **Φ**, $\mathbf{B}_d$, **H**, $\mathbf{Q}_d$ and/or **R**. Uncertainties in the plant $\mathbf{G}_d$ are treated equivalently as uncertainties in $\mathbf{Q}_d$. Admissible ranges of parameters are specified to predetermined values.

- Parameters take on discrete values. If the parameters are continuous, then they can be quantized to some reasonable finite level of resolution. Of course, quantization affects the accuracy of the filter since there will be a mismatch between the actual parameters and the assumed values. The appropriate discretization of the parameter space is an important issue of this research.

The uncertain parameters are specified by a parameter vector $\mathbf{a}_k \in \Re^P$ where $k \in \{1 \ldots K\}$ for each of the K given models and P is the dimension of the parameter vector. For each model, there will be a set of system equations, propagation equations, and filter updates similar to Equations (2.3) through (2.9) but dependent upon the assumed value of the parameters, $\mathbf{a}_k$. In accordance with the first assumption above, for each model the system and measurement equations are defined as:

$$\mathbf{x}_k(t_i) = \mathbf{\Phi}_k(t_i, t_{i-1})\mathbf{x}_k(t_{i-1}) + \mathbf{B}_{dk}(t_{i-1})\mathbf{u}(t_{i-1}) + \mathbf{G}_{dk}(t_{i-1})\mathbf{w}_{dk}(t_{i-1}) \tag{2.10}$$

$$\mathbf{z}(t_i) = \mathbf{H}_k(t_i)\mathbf{x}_k(t_i) + \mathbf{v}_k(t_i) \tag{2.11}$$

Based on the models, the corresponding Kalman propagation equations are specified as:

$$\hat{\mathbf{x}}_k(t_i^-) = \mathbf{\Phi}_k(t_i, t_{i-1})\hat{\mathbf{x}}_k(t_{i-1}^+) + \mathbf{B}_{dk}(t_{i-1})\mathbf{u}(t_{i-1}) \tag{2.12}$$

$$\mathbf{P}_k(t_i^-) = \mathbf{\Phi}_k(t_i, t_{i-1})\mathbf{P}_k(t_{i-1}^+)\mathbf{\Phi}_k^T(t_i, t_{i-1}) + \mathbf{G}_{dk}(t_{i-1})\mathbf{Q}_{dk}(t_{i-1})\mathbf{G}_{dk}^T(t_{i-1}) \tag{2.13}$$

The update equations corresponding to each model are as follows:

$$\mathbf{A}_{\mathrm{k}}(t_i) = \mathbf{H}_{\mathrm{k}}(t_i)\mathbf{P}_{\mathrm{k}}(t_i^-)\mathbf{H}_{\mathrm{k}}{}^{\mathrm{T}}(t_i) + \mathbf{R}_{\mathrm{k}}(t_i) \tag{2.14}$$

$$\mathbf{K}_{\mathrm{k}}(t_i) = \mathbf{P}_{\mathrm{k}}(t_i^-)\mathbf{H}_{\mathrm{k}}{}^{\mathrm{T}}(t_i)\mathbf{A}_{\mathrm{k}}{}^{-1}(t_i) \tag{2.15}$$

$$\hat{\mathbf{x}}_{\mathrm{k}}(t_i^+) = \hat{\mathbf{x}}_{\mathrm{k}}(t_i^-) + \mathbf{K}_{\mathrm{k}}(t_i)[\mathbf{z}_i - \mathbf{H}_{\mathrm{k}}(t_i)\hat{\mathbf{x}}_{\mathrm{k}}(t_i^-)] \tag{2.16}$$

$$\mathbf{P}_{\mathrm{k}}(t_i^+) = \mathbf{P}_{\mathrm{k}}(t_i^-) - \mathbf{K}_{\mathrm{k}}(t_i)\mathbf{H}_{\mathrm{k}}(t_i)\mathbf{P}_{\mathrm{k}}(t_i^-) \tag{2.17}$$

Before the measurement has been incorporated by the update calculation, the difference between the measurement and the best prediction of that measurement before it arrives, gives the residual. This residual is calculated for each model k ∈ {1, 2, …, K} and is given by:

$$\mathbf{r}_{\mathrm{k}}(t_i) = \mathbf{z}_i - \mathbf{H}_{\mathrm{k}}(t_i)\hat{\mathbf{x}}_{\mathrm{k}}(t_i^-) \tag{2.18}$$

Now the goal is to calculate the hypothesis conditional probability associated with each model. This probability for each of the models is actually the probability that **a** assumes the value $\mathbf{a}_{\mathrm{k}}$, conditioned on the measurement history through time t_i, and is given by:

$$p_{\mathrm{k}}(t_i) = \mathrm{Prob}[\mathbf{a} = \mathbf{a}_{\mathrm{k}} \mid \mathbf{Z}(t_i) = \mathbf{Z}_i] \tag{2.19}$$

The measurement history $\mathbf{Z}(t_i)$ is composed of partitions which are actually the measurement vectors $\mathbf{z}(t_1)\ldots\mathbf{z}(t_i)$ available at the sample times $\{t_1, \ldots, t_i\}$. The realization of the measurement vector $\mathbf{Z}_i$ is composed of the vectors of measurement data, $\mathbf{z}_1, \ldots, \mathbf{z}_i$.

It has been shown [29,35] that $p_{\mathrm{k}}(t_i)$ is evaluated recursively by

$$p_{\mathrm{k}}(t_i) = \frac{f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\mathbf{z}_i \mid \mathbf{a}_{\mathrm{k}}, \mathbf{Z}_{i-1})\, p_{\mathrm{k}}(t_{i-1})}{\sum_{\mathrm{j}=1}^{\mathrm{K}} f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\mathbf{z}_i \mid \mathbf{a}_{\mathrm{j}}, \mathbf{Z}_{i-1})\, p_{\mathrm{j}}(t_{i-1})} \tag{2.20}$$

where $p_{\mathrm{k}}(t_{i\text{-}1})$ is the corresponding conditional probability at the previous sample time. Notice that there is an inherent problem when a probability p_{k} goes to zero; the probability then goes to zero for all time. There have been ad hoc ways (through lower bounding) of preventing this from occurring that have been implemented [35,37,38,39,56,59] as well as Markov models for parameter propagation [35] that similarly preclude such *lockout* phenomena.

The conditional density function in the denominator term is given by:

$$f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\mathbf{z}_i \mid \mathbf{a}_\mathrm{k}, \mathbf{Z}_{i-1}) = \beta_\mathrm{k} \exp\{\cdot\} \tag{2.21}$$

$$\beta_\mathrm{k} = \frac{1}{(2\pi)^{m/2} \left|\mathbf{A}_\mathrm{k}(t_i)\right|^{1/2}} \tag{2.22}$$

$$\{\cdot\} = \left\{-\tfrac{1}{2}\mathbf{r}_\mathrm{k}^{\mathrm{T}}(t_i)\mathbf{A}_\mathrm{k}^{-1}(t_i)\mathbf{r}_\mathrm{k}(t_i)\right\} \tag{2.23}$$

where m is the dimension of $\mathbf{r}_\mathrm{k}(t_i)$. From Equation (2.23) it is evident that the models with the residuals that are most in consonance with their conditional covariance will have conditional density functions evaluations that yield the greatest probabilities in Equation (2.20), whereas residuals that are larger than anticipated by $\mathbf{A}_\mathrm{k}(t_i)$ yield smaller probabilities $p_\mathrm{k}(t_i)$.

The Bayesian minimum mean square error (MMSE) yields:

$$\hat{\mathbf{x}}_{\mathbf{MMAE}}(t_i^+) = \mathrm{E}\{\mathbf{x}(t_i) \mid \mathbf{Z}(t_i) = \mathbf{Z_i}\} = \sum_{\mathrm{k}=1}^{\mathrm{K}} \hat{\mathbf{x}}_\mathrm{k}(t_i^+) \cdot p_\mathrm{k}(t_i) \tag{2.24}$$

A variation to the MMSE approach is the maximum a posteriori (MAP) method that chooses the state estimate corresponding to the model that has the highest probability. This estimate is given by:

$$\hat{\mathbf{x}}_{\mathbf{MMAE-MAP}}(t_i^+) = \hat{\mathbf{x}}_\mathrm{j}(t_i^+) \text{ for } \mathrm{j} = \arg\left\{\max_\mathrm{k}[p_\mathrm{k}(t_i)]\right\} \tag{2.25}$$

The MAP approach does not perform blending, as does the MMSE approach. Thus, there is a potential with MAP to switch abruptly from one estimation model to another. Comparisons of the approaches have been demonstrated in [20,39,53,54], with minor differences in results. If the hypothesis models are separated significantly, all probability will *flow* to a single model, regardless. The differences between MAP and MMSE implementations are problem-dependent and more specifically, dependent on the filter tuning

2.3 Multiple Model Adaptive Control Development

MMAC development builds upon the theory discussed in previous section on MMAE. For each model of the plant based on a specific value of parameter vector $\mathbf{a}_{\mathrm{k}}$, a standard linear-quadratic full-state feedback regulator or any other appropriate controller can be designed. Without consideration of the need to estimate the parameters or the states, the form of the elemental full-state feedback controller for a given model, i.e. for a model based on $\mathbf{a}_{\mathrm{k}}$, is of the form:

$$\mathbf{u}_{\mathrm{k}}(t_i) = -\mathbf{G}_{\mathrm{c}}^{*}(t_i, \mathbf{a}_{\mathrm{k}})\mathbf{x}_{\mathrm{k}}(t_i) \tag{2.26}$$

Now using *assumed certainty equivalence design* [35], an adaptive controller can be designed using an analog approach to the MMAE. The MMAE of course provides the adaptive estimate of the state by blending individual filter state estimates using the conditional hypothesis probabilities. The blending of control approach is similar to the blending of the states by the MMAE. The blending of control outputs of individual LQG controllers replaces the blending of states as illustrated in Figure 1.1 from Chapter 1. Each elemental controller is designed based on the assumed parameter vector $\mathbf{a}_{\mathrm{k}}$ using an appropriate method. The general form of the resultant elemental control is:

$$\mathbf{u}_{\mathrm{k}}(t_i) = -\mathbf{G}_{\mathrm{c}}^{*}(t_i, \mathbf{a}_{\mathrm{k}})\hat{\mathbf{x}}_{\mathrm{k}}(t_i^{+}) \tag{2.27}$$

The perfectly known state vector in Equation (2.26) is replaced with a state estimate from a Kalman filter designed for a model based on $\mathbf{a}_\mathrm{k}$. A Linear Quadratic Regulator (LQR) design is used typically for $-\mathbf{G}_\mathrm{c}^*(t_i,*)$, but other implementations are possible. Stevens [60] used a command generator tracker with proportional-plus-integral (CGT/PI) control designed via LQ methods rather than a simple regulator for the elemental controllers. Now using the same conditional hypothesis probabilities as were used for MMAE, the control is calculated as:

$$\mathbf{u}_{\mathbf{MMAC}}(t_i) = \sum_{\mathrm{k}=1}^{\mathrm{K}} \mathbf{u}_\mathrm{k}(t_i)\, p_\mathrm{k}(t_i) \tag{2.28}$$

2.4 Multiple Model Adaptive Estimation Based Control Development

MMAE-based control is a variation to MMAC and in fact uses MMAE as part of its structure, as shown in Figure 1.2 from Chapter 1. Consider the case in which the parameters of the plant in the system to be controlled were known exactly; then it would still be very easy task to design a controller. If the parameters of the plant changed and the parameters were known and there was enough time to design a full-state feedback in real time, then it would still be very easy to design a control algorithm. As Miller showed that state estimation could be computed based on a parameter estimate fed into an additional single state estimator based on $\hat{\mathbf{a}}$ [44], control can also be based on parameter estimates rather than blending of controls, each based on a single hypothesized parameter value. An MMAE could determine the best parameter estimate and a controller look-up table could be designed to provide the desired feedback control. This proposed architecture is shown in Figure 1.3 in Chapter 1. For MMAE-based control, the parameters are estimated along with the states. These parameter estimates are used in the controller computation block to generate the appropriate controller, as shown in Figure 1.2.

One implementation of the MMAE-based control algorithm uses only the state estimate returned by the MMAE and a full-state feedback controller designed for a nominal set of assumed plant parameters and used for all time. This controller must be robust to parameter variations in the system, so choice of the nominal parameters is important. The controller is given by:

$$\mathbf{u}(t_i) = -\mathbf{G}_{\mathrm{c}}^{*}(t_i, \mathbf{a}_{\mathrm{nom}})\hat{\mathbf{x}}_{\mathrm{MMAE}}(t_i^{+}) \tag{2.29}$$

For the typical implementation of MMAE-based control, hypothesis conditional probability weighting is used to determine the parameter estimate as well as the state estimate. The form of the state estimate is the same as the MMAE and is given by Equation (2.24). The corresponding parameter estimate is similar and is given by

$$\hat{\mathbf{a}}_{\mathbf{MMAE}}(t_i^{+}) = \mathrm{E}\{\mathbf{a}(t_i) \mid \mathbf{Z}(t_i) = \mathbf{Z}_i\} = \sum_{\mathrm{k}=1}^{K} \mathbf{a}_{\mathrm{k}} \cdot p_{\mathrm{k}}(t_i) \tag{2.30}$$

However, Miller's approach could be used such that the parameter estimate is used to feed an additional single Kalman filter on-line to give the best state estimates.

Now, consider the evaluation of the controller based on the parameter estimate $\hat{\mathbf{a}}_{\mathbf{MMAE}}$. Conceptually, either this controller evaluation could be a complete real-time design of a full-state feedback controller by solving backward Riccati equations, or it could be accomplished through some implementation of a table look-up (possibly with interpolations) of prestored solutions. Whereas the on-line design will provide the best control for the given parameter estimate, a table look-up algorithm provides speed. Regardless of the controller evaluation, the controller gain evaluation is multiplied by the state estimate to yield the output. Of course this assumes that the state estimate is valid. The control output calculation is of the form:

$$\mathbf{u}(t_i) = -\mathbf{G}_c^*\left[t_i ; \hat{\mathbf{a}}_{\mathbf{MMAE}}(t_i)\right] \; \hat{\mathbf{x}}_{\mathbf{MMAE}}(t_i^+) \tag{2.31}$$

The benefits of the parameter estimation by MMAE are a driving factor for this adaptive control method and thus this discussion. As Miller [44] used parameter estimates from an MMAE to improve state estimation in his M^{3}AE approach, there is also the potential to improve control using the MMAE-based approach. However, Miller found that in order to obtain a performance benefit of the M^{3}AE approach, a moving-bank M^{3}AE as done by Vasquez [63,64,65] is necessary in order to force blending. It is conjectured that a similar M^{3}AE-base control approach would also require forced blending to realize full performance benefits.

As will be discussed, there is a trade-off between multiple model design for parameter estimation, state estimation, and control with respect to optimal discretization of parameter values, using the architecture in Figure 1.2. As initially conceived by Sheldon [56,57] for MMAC, there is only one discretization of the parameter space to yield the discrete values $\mathbf{a}_1 \ldots \mathbf{a}_K$ upon which to base the elemental LQG controller within the MMAC structure. As opposed to an MMAC, MMAE-based control should not suffer from that drawback since all components (filters, full-state feedback controllers, etc.) can be specified separately. However, the model placement or discretization is still an essential issue for these individual components. In this case, the placement of the models for the parameter estimator and the potential placement of the controllers in a table look-up scheme could be accomplished separately. Finally, using Miller's results [44], the state estimator could be designed on-line with the MMAE parameter estimates. The state estimate is from a single filter based on the best parameter estimate from the MMAE and hence, there is not a discretization problem for that one state estimator.

2.5 Parameter Space Discretization

Parameter space discretization entails the determination of the point values of parameters for the models that best represent the possible operating conditions of the plant. Determination of the placement of the possible parameter point value is based on the behavior of how the plant changes as its assumed parameters vary in the parameter space. A linear placement of the models would coincide with a linear variation of the uncertain parameters. This is a possibility, but does not apply to every system and so such a discretization is not always a viable option. Placement of the models based on assumed behavior of the plant tended to be intuitive and ad hoc [13,16, 17,18,19,20,37,53,54,63,64,65]. Such ad hoc procedures, though not optimal, were effective.

Sheldon's work brought a formal procedure to MMAE and MMAC design. Sheldon said it best himself [56]: "Although it seems like such an easy solution, one must remember that before this research was accomplished, there was great confusion as to what parameters to choose for the design." Based on conclusions by Matthes [32], Sheldon directed his research to use the performance of the state or parameter estimator as the criterion for optimization of the MMAE and performance of the controller as the criterion for the optimization of the MMAC. Thus, there is a different optimization approach for state estimation, parameter estimation, and control. The different optimizations yield different results, and so the model placement will be different for state estimation, parameter estimation and control.

Sheldon's approach to optimal parameter placement for state estimation is to minimize a cost function composed of the average value (over the parameter space) of the mean squared estimation error. The average is taken over the range of the actual possible values of the parameters. This cost function is expressed as:

$$\mathrm{J}_{2\,\hat{\mathrm{x}}} \equiv \frac{\int_A \mathrm{E}\{[\mathbf{x}-\hat{\mathbf{x}}]^{\mathrm{T}}\mathbf{W}[\mathbf{x}-\hat{\mathbf{x}}]\}da}{\int_A da} \tag{2.32}$$

where

$$\int_A da \equiv \int_{A_P}\cdots\int_{A_2}\int_{A_1} da_1\, da_2 \ldots da_P \tag{2.33}$$

provides the necessary scaling for $A_j \subset A$, $j = \{1,\ldots, P\}$ which specifies the P dimensional parameter space, and **W** is the designer-specified weights for the states. Similarly, the cost function for optimal model placement for parameter estimation is defined as:

$$\mathrm{J}_{2\,\mathrm{a}} \equiv \frac{\int_A \mathrm{E}\{[\mathbf{a}-\hat{\mathbf{a}}]^{\mathrm{T}}\mathbf{W}[\mathbf{a}-\hat{\mathbf{a}}]\}da}{\int_A da} \tag{2.34}$$

For regulation, the cost function minimized is the output squared over the range of possible values of the parameters and is expressed as:

$$\mathrm{J}_{2\,\mathrm{c}} \equiv \frac{\int_A \mathrm{E}\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y}\}da}{\int_A da} \tag{2.35}$$

where **y**=**Cx** is the output to be regulated, expressed as a linear function of the states.

Sheldon used a 5-step design procedure to approximate and to minimize the appropriate cost function numerically [56]. The steps are summarized as follows:

1) Describe in terms of the parameter vector **a**, the truth model of the system and the filter.
2) Choose the number of filters *K* in the MMAE or MMAC.
3) Choose a representative parameter set, $\{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_K\}$ to begin the minimization.
4) Use a numerical integration technique to evaluate $J_{2\hat{x}}$ (or J_{2a} or J_{2c}). This evaluation depends on determining to which filter the MMAE/MMAC will converge. Determination of the convergence is discussed in Section 2.5.3.
5) Use a vector minimization technique with the functional evaluation from Step (4) to minimize $J_{2\hat{x}}$ (or J_{2a} or J_{2c}).

2.5.1 Autocorrelation of the State Estimation Error

In order to discretize the parameter space of the MMAE, it is necessary to evaluate $J_{2\hat{x}}$, which in turn requires an evaluation of $E\{[\mathbf{x}-\hat{\mathbf{x}}]^T\mathbf{W}[\mathbf{x}-\hat{\mathbf{x}}]\}$. The derivation of this expression begins with the development of the autocorrelation of the states of the true system augmented with the filter states. This derivation follows [56].

Since the MMAE is a bank of Kalman filters, each of the K filters in the bank is based on a model of the system that is assumed to be correct by that filter. The system model on which the kth possible filter (k = 1,…, K) is based is:

$$\mathbf{x}_k(t_i) = \mathbf{\Phi}_k(t_i, t_{i-1})\mathbf{x}_k(t_{i-1}) + \mathbf{G}_{dk}(t_{i-1})\mathbf{w}_{dk}(t_{i-1}) \tag{2.36}$$

with measurement:

$$\mathbf{z}(t_i) = \mathbf{H}_k(t_i)\mathbf{x}_k(t_i) + \mathbf{v}_k(t_i) \tag{2.37}$$

The truth model is expressed as:

$$\mathbf{x}_t(t_i) = \mathbf{\Phi}_t(t_i, t_{i-1})\mathbf{x}_t(t_{i-1}) + \mathbf{G}_{dt}(t_{i-1})\mathbf{w}_{dt}(t_{i-1}) \tag{2.38}$$

with measurement

$$\mathbf{z}(t_i) = \mathbf{H}_{\mathrm{t}}(t_i)\mathbf{x}_{\mathrm{t}}(t_i) + \mathbf{v}_{\mathrm{t}}(t_i) \tag{2.39}$$

In subsequent discussions and development, it is assumed that the filters and the system are at steady state. Thus, the following assumptions apply:

$$t_{i+1}\text{-}t_i = \text{a constant } \forall\, i$$

$$\mathbf{\Phi}_{\mathrm{k}}(t_{i+1},t_i) = \mathbf{\Phi}_{\mathrm{k}} \text{ and } \mathbf{\Phi}_{\mathrm{t}}(t_{i+1},t_i) = \mathbf{\Phi}_{\mathrm{t}}\ \forall\, t_{i+1},t_i$$

$$\mathbf{H}_{\mathrm{k}}\,(t_i) = \mathbf{H}_{\mathrm{k}} \text{ and } \mathbf{H}_{\mathrm{t}}\,(t_i) = \mathbf{H}_{\mathrm{t}}\ \forall\, t_i$$

$$\mathbf{G}_{\mathrm{dk}}(t_i) = \mathbf{G}_{\mathrm{dk}} \text{ and } \mathbf{G}_{\mathrm{dt}}(t_i) = \mathbf{G}_{\mathrm{dt}} \forall\, t_i$$

It is further assumed that the filters are constant-gain algorithms and so:

$$\mathbf{K}_{\mathrm{k}}\,(t_i) = \mathbf{K}_{\mathrm{k}}\ \forall\, t_i$$

The development of the equations for the filter models follows that of the Kalman filter. For the k$^{\mathrm{th}}$ filter, the propagation equation of the state in the Kalman filter is:

$$\hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^-) = \mathbf{\Phi}_{\mathrm{k}}\hat{\mathbf{x}}_{\mathrm{k}}(t_i^+) \tag{2.40}$$

and the measurement update at time t_i is:

$$\hat{\mathbf{x}}_{\mathrm{k}}(t_i^+) = \hat{\mathbf{x}}_{\mathrm{k}}(t_i^-) + \mathbf{K}_{\mathrm{k}}[\mathbf{z}(t_i) - \mathbf{H}_{\mathrm{k}}\hat{\mathbf{x}}_{\mathrm{k}}(t_i^-)] \tag{2.41}$$

The measurement is taken as:

$$\mathbf{z}(t_i) = \mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) + \mathbf{v}_{\mathrm{t}}(t_i) \tag{2.42}$$

Now substitute Equation (2.42) into Equation (2.41) to yield the filter state estimate just after the measurement update at t_i^+.

$$\begin{aligned}\hat{\mathbf{x}}_{\mathrm{k}}(t_i^+) &= \hat{\mathbf{x}}_{\mathrm{k}}(t_i^-) + \mathbf{K}_{\mathrm{k}}[\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) + \mathbf{v}_{\mathrm{t}}(t_i) - \mathbf{H}_{\mathrm{k}}\hat{\mathbf{x}}_{\mathrm{k}}(t_i^-)] \\ &= (\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\hat{\mathbf{x}}_{\mathrm{k}}(t_i^-) + \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) + \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}}\mathbf{v}_{\mathrm{t}}(t_i)\end{aligned} \tag{2.43}$$

In order to derive an expression for state estimates of the kth filter at t_{i+1}^{-}, substitute the measurement update into Equation (2.40) which yields:

$$\begin{aligned}\hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^{-}) &= \mathbf{\Phi}_{\mathrm{k}}\left(\hat{\mathbf{x}}_{\mathrm{k}}(t_i^{-}) + \mathbf{K}_{\mathrm{k}}[\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) + \mathbf{v}_{\mathrm{t}}(t_i) - \mathbf{H}_{\mathrm{k}}\hat{\mathbf{x}}_{\mathrm{k}}(t_i^{-})]\right) \\ &= \mathbf{\Phi}_{\mathrm{k}}(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\hat{\mathbf{x}}_{\mathrm{k}}(t_i^{-}) + \mathbf{\Phi}_{\mathrm{k}}\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) + \mathbf{\Phi}_{\mathrm{k}}\mathbf{K}_{\mathrm{k}}\mathbf{v}_{\mathrm{t}}(t_i)\end{aligned} \tag{2.44}$$

The state equation for the true system does not have a measurement update and is:

$$\mathbf{x}_{\mathrm{t}}(t_{i+1}) = \mathbf{\Phi}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) + \mathbf{G}_{\mathrm{dt}}\mathbf{w}_{\mathrm{dt}}(t_i) \tag{2.45}$$

Equation (2.44) and Equation (2.45) are combined to form the expression for the augmented system and filter state equations:

$$\begin{aligned}\begin{bmatrix}\mathbf{x}_{\mathrm{t}}(t_{i+1}) \\ \hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^{-})\end{bmatrix} &= \begin{bmatrix}\mathbf{\Phi}_{\mathrm{t}} & \mathbf{0} \\ \mathbf{\Phi}_{\mathrm{k}}\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}} & \mathbf{\Phi}_{\mathrm{k}}(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\end{bmatrix}\begin{bmatrix}\mathbf{x}_{\mathrm{t}}(t_i) \\ \hat{\mathbf{x}}_{\mathrm{k}}(t_i^{-})\end{bmatrix} \\ &+ \begin{bmatrix}\mathbf{0} \\ \mathbf{\Phi}_{\mathrm{k}}\mathbf{K}_{\mathrm{k}}\end{bmatrix}\mathbf{v}_{\mathrm{t}}(t_i) + \begin{bmatrix}\mathbf{G}_{\mathrm{dt}} \\ \mathbf{0}\end{bmatrix}\mathbf{w}_{\mathrm{dt}}(t_i)\end{aligned} \tag{2.46}$$

Now define

$$\mathbf{Y} = \begin{bmatrix}\mathbf{\Phi}_{\mathrm{t}} & \mathbf{0} \\ \mathbf{\Phi}_{\mathrm{k}}\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}} & \mathbf{\Phi}_{\mathrm{k}}(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\end{bmatrix} \tag{2.47}$$

and

$$\mathbf{G}_o = \begin{bmatrix}\mathbf{G}_{\mathrm{dt}} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Phi}_{\mathrm{k}}\mathbf{K}_{\mathrm{k}}\end{bmatrix} \tag{2.48}$$

Also, note that the statistics for the modeled noise are given as:

$$\mathrm{E}\left\{\begin{bmatrix}\mathbf{w}_{\mathrm{dt}}(t_i) \\ \mathbf{v}_{\mathrm{t}}(t_i)\end{bmatrix}\begin{bmatrix}\mathbf{w}_{\mathrm{dt}}(t_j)^{\mathrm{T}} & \mathbf{v}_{\mathrm{t}}(t_j)^{\mathrm{T}}\end{bmatrix}\right\} = \begin{cases}\mathbf{Q_0}(t_i) & t_i = t_j \\ \mathbf{0} & t_i \neq t_j\end{cases} \tag{2.49}$$

$$\mathbf{Q_0}(t_i) = \begin{bmatrix}\mathbf{Q}_{\mathrm{dt}}(t_i) & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{\mathrm{t}}(t_i)\end{bmatrix} \tag{2.50}$$

The autocorrelation of the augmented state equations can be expressed now as:

$$E\left\{\begin{bmatrix}\mathbf{x}_t(t_{i+1})\\ \hat{\mathbf{x}}_k(t_{i+1}^-)\end{bmatrix}\begin{bmatrix}\mathbf{x}_t(t_{i+1})^T & \hat{\mathbf{x}}_k(t_{i+1}^-)^T\end{bmatrix}\right\} = \mathbf{\Xi}_k(t_{i+1}^-) = \mathbf{Y}\mathbf{\Xi}_k(t_i^-)\mathbf{Y}^T + \mathbf{G_0 Q_0 G_0}^T \tag{2.51}$$

Assuming that $\mathbf{Y}$ is a contraction, $\mathbf{\Xi}_k(t_i)$ approaches a constant value as t_i approaches infinity. This constant value is denoted as $\mathbf{\Xi}_k$. The lower right partition of $\mathbf{\Xi}_k$ is $E\{[\hat{\mathbf{x}}][\hat{\mathbf{x}}]^T\}$. However, rather than the autocorrelation of the state estimation, the cost function in Equation (2.32) requires the autocorrelation of the difference between the state estimate and the true state. Assuming that the filter design model may be different from the true system model, the following transformation is required:

$$\tilde{\mathbf{x}}_k = \hat{\mathbf{x}}_k - \mathbf{T}\mathbf{x}_t \tag{2.52}$$

$\mathbf{T}$ is a transformation matrix that allows the state estimates of the kth filter to be of different dimension from the true system dimension. With the substitution of Equation (2.52), the augmented system becomes:

$$\begin{bmatrix}\mathbf{x}_t(t_{i+1})\\ \tilde{\mathbf{x}}_k(t_{i+1}^-)\end{bmatrix} = \begin{bmatrix}\mathbf{\Phi}^t & \mathbf{0}\\ \mathbf{T\Delta\Phi} - \mathbf{\Phi}_k\mathbf{K}_k\mathbf{\Delta H} & \mathbf{\Phi}_k(\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\end{bmatrix}\begin{bmatrix}\mathbf{x}_t(t_i)\\ \tilde{\mathbf{x}}_k(t_i^-)\end{bmatrix} + \begin{bmatrix}\mathbf{0}\\ \mathbf{\Phi}_k\mathbf{K}_k\end{bmatrix}\mathbf{v}_t(t_i) + \begin{bmatrix}\mathbf{G}_{dt}\\ -\mathbf{TG}_{dt}\end{bmatrix}\mathbf{w}_{dt}(t_i) \tag{2.53}$$

where

$$\mathbf{T\Delta\Phi} \equiv \mathbf{\Phi}_k\mathbf{T} - \mathbf{T\Phi}_t \tag{2.54}$$

$$\mathbf{\Delta H} \equiv \mathbf{H}_k\mathbf{T} - \mathbf{H}_t \tag{2.55}$$

A more in-depth derivation can be found in [56]. Now define

$$\mathbf{Y'} = \begin{bmatrix}\mathbf{\Phi}_t & \mathbf{0}\\ \mathbf{T\Delta\Phi} - \mathbf{\Phi}_k\mathbf{K}_k\mathbf{\Delta H} & \mathbf{\Phi}_k(\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\end{bmatrix} \tag{2.56}$$

and

$$\mathbf{G}'_o = \begin{bmatrix} \mathbf{G}_{\mathrm{dt}} & \mathbf{0} \\ -\mathbf{T}\mathbf{G}_{\mathrm{dt}} & \mathbf{\Phi}_{\mathrm{k}}\mathbf{K}_{\mathrm{k}} \end{bmatrix} \tag{2.57}$$

Also, note that the statistics for the modeled noise is given as in Equations (2.49).

The autocorrelation of the augmented state equations can be expressed now as:

$$\mathrm{E}\left\{ \begin{bmatrix} \mathbf{x}_{\mathrm{t}}(t_{i+1}) \\ \tilde{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^{-}) \end{bmatrix} \begin{bmatrix} \mathbf{x}_{\mathrm{t}}(t_{i+1})^{\mathrm{T}} & \tilde{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^{-})^{\mathrm{T}} \end{bmatrix} \right\} = \mathbf{\Gamma}_{\mathrm{k}}(t_{i+1}^{-}) = \mathbf{Y}'\mathbf{\Gamma}_{\mathrm{k}}(t_i^{-})\mathbf{Y}'^{\mathrm{T}} + \mathbf{G}'_{\mathbf{0}}\mathbf{Q}_{\mathbf{0}}\mathbf{G}'^{\mathrm{T}}_{\mathbf{0}} \tag{2.58}$$

Assuming that $\mathbf{Y}'$ is a contraction, $\mathbf{\Gamma}_{\mathrm{k}}(t_i)$ approaches a constant value as t_i approaches infinity. This constant value is denoted as $\mathbf{\Gamma}_{\mathrm{k}}$. The lower right partition of $\mathbf{\Gamma}_{\mathrm{k}}$ is $\mathrm{E}\{[\tilde{\mathbf{x}}]\,[\tilde{\mathbf{x}}]^{\mathrm{T}}\}$ or $\mathrm{E}\{[\mathbf{x}-\hat{\mathbf{x}}]\,[\mathbf{x}-\hat{\mathbf{x}}]^{\mathrm{T}}\}$ for the selected filter. Note that for Equation (2.32), $\mathrm{E}\{[\mathbf{x}-\hat{\mathbf{x}}]^{\mathrm{T}}\,\mathbf{W}\,[\mathbf{x}-\hat{\mathbf{x}}]\}$ equals $\mathrm{tr}[\mathbf{W}\,\mathrm{E}\{[\tilde{\mathbf{x}}]\,[\tilde{\mathbf{x}}]^{\mathrm{T}}\}]$. Subsequent sections will discuss how the selected filter is determined.

2.5.2 Autocorrelation of the Regulator Output

Essential to the discretization of the MMAC is the evaluation of Equation (2.35) for the defined parameter space. To perform this evaluation, the equations for the output autocorrelation have to be developed. For the MMAC, the expectation of the regulation output autocorrelation (mean squared regulation error) given by:

$$\begin{aligned} \mathrm{E}\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y}\} &\equiv \mathrm{E}\{\mathrm{tr}(\mathbf{W}\mathbf{y}\mathbf{y}^{\mathrm{T}})\} \\ &\equiv \mathrm{tr}(\mathbf{W}\mathrm{E}\{\mathbf{C}\mathbf{x}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}^{\mathrm{T}}\mathbf{C}^{\mathrm{T}}\}) \\ &\equiv \mathrm{tr}(\mathbf{W}\mathbf{C}\mathbf{\Psi}_{\mathbf{x}_{\mathrm{t}}}\mathbf{C}^{\mathrm{T}}) \end{aligned} \tag{2.59}$$

where $\mathbf{\Psi}_{\mathbf{x}_{\mathrm{t}}} = \mathrm{E}\{[\mathbf{x}_{\mathrm{t}}][\mathbf{x}_{\mathrm{t}}]^{\mathrm{T}}\}$. Now it is a matter of finding the expression for the autocorrelation of the true states. The following derivation follows that of [56,57] and stresses the relevant details for this research.

For the MMAE, note that in Equation (2.45), since there are no feedback terms, the filter does not affect the true states. Likewise, there are no feedback terms in the filter state equations as shown in Equation (2.40). However, for the MMAC, the model based on $\mathbf{a}_\mathrm{k}$ is given by:

$$\mathbf{x}_\mathrm{k}(t_{i+1}) = \mathbf{\Phi}_\mathrm{k}(t_{i+1},t_i)\mathbf{x}_\mathrm{k}(t_i) + \mathbf{B}_{\mathrm{dk}}(t_i)\mathbf{u}_\mathrm{k}(t_i) + \mathbf{G}_{\mathrm{dk}}(t_i)\mathbf{w}_{\mathrm{dk}}(t_i) \tag{2.60}$$

$$\mathbf{y}_\mathrm{k}(t_i) = \mathbf{C}_\mathrm{k}(t_i)\mathbf{x}_\mathrm{k}(t_i)$$

The true system is modeled by

$$\mathbf{x}_\mathrm{t}(t_{i+1}) = \mathbf{\Phi}_\mathrm{t}(t_{i+1},t_i)\mathbf{x}_\mathrm{t}(t_i) + \mathbf{B}_{\mathrm{dt}}(t_i)\mathbf{u}_\mathrm{t}(t_i) + \mathbf{G}_{\mathrm{dt}}(t_i)\mathbf{w}_{\mathrm{dt}}(t_i) \tag{2.61}$$

In subsequent discussions and development, it is assumed that the true and design models are time-invariant. Thus, the following assumptions apply:

$$t_{i+1} - t_i = \text{a constant } \forall\, i$$

$$\mathbf{\Phi}_\mathrm{k}(t_{i+1},t_i) = \mathbf{\Phi}_\mathrm{k} \text{ and } \mathbf{\Phi}_\mathrm{t}(t_{i+1},t_i) = \mathbf{\Phi}_\mathrm{t}\ \forall\, t_{i+1}, t_i$$

$$\mathbf{H}_\mathrm{k}(t_i) = \mathbf{H}_\mathrm{k} \text{ and } \mathbf{H}_\mathrm{t}(t_i) = \mathbf{H}_\mathrm{t}\ \forall\, t_i$$

$$\mathbf{C}_\mathrm{k}(t_i) = \mathbf{C}_\mathrm{k}\ \forall\, t_i$$

$$\mathbf{G}_{\mathrm{dk}}(t_i) = \mathbf{G}_{\mathrm{dk}} \text{ and } \mathbf{G}_{\mathrm{dt}}(t_i) = \mathbf{G}_{\mathrm{dt}}\ \forall\, t_i$$

$$\mathbf{B}_{\mathrm{dk}}(t_i) = \mathbf{B}_{\mathrm{dk}} \text{ and } \mathbf{B}_{\mathrm{dt}}(t_i) = \mathbf{B}_{\mathrm{dt}}\ \forall\, t_i$$

It is further assumed that the filters and controllers are constant-gain steady-state algorithms, and so:

$$\mathbf{K}_\mathrm{k}(t_i) = \mathbf{K}_\mathrm{k}\ \forall\, t_i$$

$$\mathbf{G}^*_{\mathrm{ck}}(t_i) = \mathbf{G}^*_{\mathrm{ck}}\ \forall\, t_i$$

For the kth filter, the propagation equation of the state in the Kalman filter is:

$$\hat{\mathbf{x}}_\mathrm{k}(t_{i+1}^-) = \mathbf{\Phi}_\mathrm{k}\hat{\mathbf{x}}_\mathrm{k}(t_i^+) + \mathbf{B}_{\mathrm{dk}}\mathbf{u}_\mathrm{k}(t_i) \tag{2.62}$$

and the measurement update at time t_i is the same as the MMAE:

$$\hat{\mathbf{x}}_{\mathrm{k}}(t_i^+) = \hat{\mathbf{x}}_{\mathrm{k}}(t_i^-) + \mathbf{K}_{\mathrm{k}}[\mathbf{z}(t_i) - \mathbf{H}_{\mathrm{k}}\hat{\mathbf{x}}_{\mathrm{k}}(t_i^-)] \tag{2.63}$$

The constant-gain state feedback controllers for each model are implemented in the form of:

$$\mathbf{u}_{\mathrm{k}}(t_i) = -\mathbf{G}^*_{\mathrm{ck}}\hat{\mathbf{x}}_{\mathrm{k}}(t_i^+) \tag{2.64}$$

Now it was assumed that the MMAC has converged to a single selected filter and that the control is given by the gain corresponding to the selected filter. Hence, the control for the truth model corresponds to the control of the k$^{\mathrm{th}}$ filter selected by convergence in probability to 1. So we have

$$p_{sel} = 1 \quad \Rightarrow \quad \mathbf{u}_{\mathrm{t}}(t_i) = \mathbf{u}_{\mathrm{k}}(t_i) \tag{2.65}$$

and by substitution, the control is given by

$$\mathbf{u}_{\mathrm{t}}(t_i) = -\mathbf{G}^*_{\mathrm{ck}}\hat{\mathbf{x}}_{\mathrm{k}}(t_i^+) \tag{2.66}$$

Now substitute the control into the Kalman filter propagation equation to yield:

$$\begin{aligned}\hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^-) &= \mathbf{\Phi}_{\mathrm{k}}\hat{\mathbf{x}}_{\mathrm{k}}(t_i^+) - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}}\hat{\mathbf{x}}_{\mathrm{k}}(t_i^+) \\ &= \left(\mathbf{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}}\right)\hat{\mathbf{x}}_{\mathrm{k}}(t_i^+)\end{aligned} \tag{2.67}$$

As done in the previous development of the MMAE, substitute the measurement update given in Equation (2.43) for the Kalman filter at t_i^+. This final substitution yields the state estimate for the k$^{\mathrm{th}}$ filter at t_{i+1}^- and is given by:

$$\begin{aligned}\hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^-) &= \left(\mathbf{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}}\right)\left\{\left(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}}\right)\hat{\mathbf{x}}_{\mathrm{k}}(t_i^-) + \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) + \mathbf{K}_{\mathrm{k}}\mathbf{v}_{\mathrm{t}}(t_i)\right\} \\ &= \left(\mathbf{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}}\right)\left(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}}\right)\hat{\mathbf{x}}_{\mathrm{k}}(t_i^-) + \left(\mathbf{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}}\right)\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) \\ &\quad + \left(\mathbf{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}}\right)\mathbf{K}_{\mathrm{k}}\mathbf{v}_{\mathrm{t}}(t_i)\end{aligned} \tag{2.68}$$

The true system is modeled by

$$\mathbf{x}_{\mathrm{t}}(t_{i+1}) = \mathbf{\Phi}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) + \mathbf{B}_{\mathrm{dt}}\mathbf{u}_{\mathrm{t}}(t_i) + \mathbf{G}_{\mathrm{dt}}\mathbf{w}_{\mathrm{dt}}(t_i) \tag{2.69}$$

Now substitute the control from Equation (2.66) and the measurement from Equation (2.44) to yield:

$$\begin{aligned}\mathbf{x}_{\mathrm{t}}(t_{i+1}) &= \mathbf{\Phi}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) - \mathbf{G}^*_{\mathrm{ck}}\mathbf{B}_{\mathrm{dt}}\left\{\hat{\mathbf{x}}_{\mathrm{k}}(t_i^-) + \mathbf{K}_{\mathrm{k}}[\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) + \mathbf{v}_{\mathrm{t}}(t_i) - \mathbf{H}_{\mathrm{k}}\hat{\mathbf{x}}_{\mathrm{k}}(t_i^-)]\right\} \\ &\quad + \mathbf{G}_{\mathrm{dt}}\mathbf{w}_{\mathrm{dt}}(t_i) \\ &= \left(\mathbf{\Phi}_{\mathrm{t}} - \mathbf{G}^*_{\mathrm{ck}}\mathbf{B}_{\mathrm{dt}}\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}}\right)\mathbf{x}_{\mathrm{t}}(t_i) - \mathbf{G}^*_{\mathrm{ck}}\mathbf{B}_{\mathrm{dt}}\left(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}}\right)\hat{\mathbf{x}}_{\mathrm{k}}(t_i^-) \\ &\quad - \mathbf{G}^*_{\mathrm{ck}}\mathbf{B}_{\mathrm{dt}}\mathbf{K}_{\mathrm{k}}\mathbf{v}^{\mathrm{t}}(t_i) + \mathbf{G}_{\mathrm{dt}}\mathbf{w}_{\mathrm{dt}}(t_i)\end{aligned} \tag{2.70}$$

The augmented system's description is given by

$$\begin{aligned}\begin{bmatrix}\mathbf{x}_{\mathrm{t}}(t_{i+1}) \\ \hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^-)\end{bmatrix} &= \begin{bmatrix}(\mathbf{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{dt}}\mathbf{G}^*_{\mathrm{ck}}\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}}) & -\mathbf{B}_{\mathrm{dt}}\mathbf{G}^*_{\mathrm{ck}}(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}}) \\ (\mathbf{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}})\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}} & (\mathbf{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}})(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\end{bmatrix}\begin{bmatrix}\mathbf{x}_{\mathrm{t}}(t_i) \\ \hat{\mathbf{x}}_{\mathrm{k}}(t_i^-)\end{bmatrix} \\ &\quad + \begin{bmatrix}-\mathbf{B}_{\mathrm{dt}}\mathbf{G}^*_{\mathrm{ck}}\mathbf{K}_{\mathrm{k}} \\ (\mathbf{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}})\mathbf{K}_{\mathrm{k}}\end{bmatrix}\mathbf{v}_{\mathrm{t}}(t_i) + \begin{bmatrix}\mathbf{G}_{\mathrm{dt}} \\ \mathbf{0}\end{bmatrix}\mathbf{w}_{\mathrm{dt}}(t_i)\end{aligned} \tag{2.71}$$

Now define

$$\mathbf{T} = \begin{bmatrix}(\mathbf{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{dt}}\mathbf{G}^*_{\mathrm{ck}}\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}}) & -\mathbf{B}_{\mathrm{dt}}\mathbf{G}^*_{\mathrm{ck}}(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}}) \\ (\mathbf{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}})\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}} & (\mathbf{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}})(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\end{bmatrix} \tag{2.72}$$

$$\mathbf{L} = \begin{bmatrix}\mathbf{G}_{\mathrm{dt}} & -\mathbf{B}_{\mathrm{dt}}\mathbf{G}^*_{\mathrm{ck}}\mathbf{K}_{\mathrm{k}} \\ \mathbf{0} & (\mathbf{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}})\mathbf{K}_{\mathrm{k}}\end{bmatrix} \tag{2.73}$$

and the statistics for the noise as:

$$\mathrm{E}\left\{\begin{bmatrix}\mathbf{w}_{\mathrm{dt}}(t_i) \\ \mathbf{v}_{\mathrm{t}}(t_i)\end{bmatrix}\begin{bmatrix}\mathbf{w}_{\mathrm{dt}}(t_j)^{\mathrm{T}} & \mathbf{v}_{\mathrm{t}}(t_j)^{\mathrm{T}}\end{bmatrix}\right\} = \begin{cases}\mathbf{Q_0}(t_i) & t_i = t_j \\ \mathbf{0} & t_i \neq t_j\end{cases} \tag{2.74}$$

where

$$\mathbf{Q_0}(t_i) = \begin{bmatrix}\mathbf{Q}_{\mathrm{dt}}(t_i) & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{\mathrm{t}}(t_i)\end{bmatrix} \tag{2.75}$$

The output autocorrelation of the augmented system at time t_{i+1} can be written conveniently as:

$$\mathrm{E}\left\{\begin{bmatrix}\mathbf{x}_{\mathrm{t}}(t_{i+1})\\ \hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^{-})\end{bmatrix}\begin{bmatrix}\mathbf{x}_{\mathrm{t}}(t_{i+1})^{\mathrm{T}} & \hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^{-})^{\mathrm{T}}\end{bmatrix}\right\} = \boldsymbol{\Xi}_{\mathrm{k}}(t_{i+1}^{-}) = \mathbf{T}\,\boldsymbol{\Xi}_{\mathrm{k}}(t_{i}^{-})\,\mathbf{T}^{\mathrm{T}} + \mathbf{L}\mathbf{Q}_{\mathbf{0}}\mathbf{L}^{\mathrm{T}} \qquad (2.76)$$

The upper left quadrant of this expression is the output autocorrelation, $\boldsymbol{\Psi}_{\mathbf{x}_{\mathrm{t}}} = \mathrm{E}\{[\mathbf{x}_{\mathrm{t}}][\mathbf{x}_{\mathrm{t}}]^{\mathrm{T}}\}$ as required in Equation (2.59) for the cost evaluation of Equation (2.35). The lower right quadrant is the autocorrelation of the state estimates of the kth filter, $\boldsymbol{\Psi}_{\hat{\mathrm{x}}_{\mathrm{k}}}$. $\boldsymbol{\Psi}_{\hat{\mathrm{x}}_{\mathrm{k}}}$ does not factor directly into the cost evaluation, but does affect the determination of which filter is active at steady state. Filter selection at steady state is discussed in the next section and in Chapter 4.

2.5.3 Lower Bound on Control for Autocorrelation of the Regulator Output

It is evident from the recursive nature of Equation (2.23) that, when one filter has *absorbed all the available probability*, then the control becomes locked onto that one controller element and away from all others. In order to negate the effects of this controller lock-out, the MMAC can be designed with an assumed artificial lower bound on the probability hypothesis computation. To implement this lower bound, assume all the filters will have a probability of $\mathrm{p}_{\min}$, except for one filter. That filter will have a probability expressed as:

$$p_{\mathrm{sel}} = 1 - p_{\min}(\mathrm{K}-1) \qquad (2.77)$$

where *sel* refers to the *selected* filter that has assumed all available probability. Now of course this lower bound will affect the placement of controller-assumed parameter values in parameter space and thus must be incorporated into the MMAC design. The derivation of the autocorrelation equations for implementing lower bounding follows the results from Sheldon [56,57].

The design for the MMAC, taking into account lower bounding, removes the assumption that only one filter will be *selected* at steady state. Each probability weight computed by the conditional hypothesis may have a nonzero value which must be factored into the position autocorrelation equations. Thus, to compute the control, the probability is assigned to each controller element as:

$$\mathbf{u}_{\mathrm{t}}(t_i) = -p_1\mathbf{G}^*_{\mathrm{c}1}\hat{\mathbf{x}}_1(t_i^+) - p_2\mathbf{G}^*_{\mathrm{c}2}\hat{\mathbf{x}}_2(t_i^+)\ \ldots\ - p_{\mathrm{K}}\mathbf{G}^*_{\mathrm{cK}}\hat{\mathbf{x}}_{\mathrm{K}}(t_i^+) \tag{2.78}$$

Since this control is input to the filters in the bank of filters, the state equations will be expressed as:

$$\hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^-) = \mathbf{\Phi}_{\mathrm{k}}\hat{\mathbf{x}}_{\mathrm{k}}(t_i^+) - \mathbf{B}_{\mathrm{d\,k}}\left(p_1\mathbf{G}^*_{\mathrm{c}1}\hat{\mathbf{x}}_1(t_i^+) + p_2\mathbf{G}^*_{\mathrm{c}2}\hat{\mathbf{x}}_2(t_i^+)\ \ldots\ + p_{\mathrm{K}}\mathbf{G}^*_{\mathrm{cK}}\hat{\mathbf{x}}_{\mathrm{K}}(t_i^+)\right) \tag{2.79}$$

and the filter update equation is given as:

$$\hat{\mathbf{x}}_{\mathrm{k}}(t_i^+) = \hat{\mathbf{x}}_{\mathrm{k}}(t_i^-) + \mathbf{K}_{\mathrm{k}}[\mathbf{z}(t_i) - \mathbf{H}_{\mathrm{k}}\hat{\mathbf{x}}_{\mathrm{k}}(t_i^-)] \tag{2.80}$$

Now substitute Equation (2.80) into Equation (2.79) for not only the k^th possible filter, but every other filter as well, to yield:

$$\begin{aligned}\hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^-) = \mathbf{\Phi}_{\mathrm{k}}&\left((\mathbf{I}-\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\hat{\mathbf{x}}_{\mathrm{k}}(t_{\mathrm{i}}^-) + \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{\mathrm{i}}) + \mathbf{K}_{\mathrm{k}}\mathbf{v}_{\mathrm{t}}(t_{\mathrm{i}})\right)\\ &- p_1\mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^*_{\mathrm{c}1*}\left((\mathbf{I}-\mathbf{K}_1\mathbf{H}_1)\hat{\mathbf{x}}_1(t_{\mathrm{i}}^-) + \mathbf{K}_1\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{\mathrm{i}}) + \mathbf{K}_1\mathbf{v}_{\mathrm{t}}(t_{\mathrm{i}})\right)\\ &- p_2\mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^*_{\mathrm{c}2}\left((\mathbf{I}-\mathbf{K}_2\mathbf{H}_2)\hat{\mathbf{x}}_2(t_{\mathrm{i}}^-) + \mathbf{K}_2\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{\mathrm{i}}) + \mathbf{K}_2\mathbf{v}_{\mathrm{t}}(t_{\mathrm{i}})\right)\ldots\\ &- p_{\mathrm{K}}\mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^*_{\mathrm{cK}}\left((\mathbf{I}-\mathbf{K}_{\mathrm{K}}\mathbf{H}_{\mathrm{K}})\hat{\mathbf{x}}_{\mathrm{K}}(t_{\mathrm{i}}^-) + \mathbf{K}_{\mathrm{K}}\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{\mathrm{i}}) + \mathbf{K}_{\mathrm{K}}\mathbf{v}_{\mathrm{t}}(t_{\mathrm{i}})\right)\end{aligned} \tag{2.81}$$

Further simplification yields:

$$\begin{aligned}\hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^-) = \mathbf{\Phi}_{\mathrm{k}}&(\mathbf{I}-\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\hat{\mathbf{x}}_{\mathrm{k}}(t_{\mathrm{i}}^-) - p_1\mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^*_{\mathrm{c}1}(\mathbf{I}-\mathbf{K}_1\mathbf{H}_1)\hat{\mathbf{x}}_1(t_{\mathrm{i}}^-)\\ &- p_2\mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^*_{\mathrm{c}2}(\mathbf{I}-\mathbf{K}_2\mathbf{H}_2)\hat{\mathbf{x}}_2(t_{\mathrm{i}}^-)\ \ldots\ - p_{\mathrm{K}}\mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^*_{\mathrm{cK}}(\mathbf{I}-\mathbf{K}_{\mathrm{K}}\mathbf{H}_{\mathrm{K}})\hat{\mathbf{x}}_{\mathrm{K}}(t_{\mathrm{i}}^-)\\ &+\left(\mathbf{\Phi}_{\mathrm{k}}\mathbf{K}_{\mathrm{k}} - \mathbf{B}_{\mathrm{d\,k}}\sum_{\mathrm{j}=1}^{\mathrm{K}} p_{\mathrm{j}}\mathbf{G}^*_{\mathrm{c\,j}}\mathbf{K}_{\mathrm{j}}\right)\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{\mathrm{i}}) + \left(\mathbf{\Phi}_{\mathrm{k}}\mathbf{K}_{\mathrm{k}} - \mathbf{B}_{\mathrm{d\,k}}\sum_{\mathrm{j}=1}^{\mathrm{K}} p_{\mathrm{j}}\mathbf{G}^*_{\mathrm{c\,j}}\mathbf{K}_{\mathrm{j}}\right)\mathbf{v}_{\mathrm{t}}(t_{\mathrm{i}})\end{aligned} \tag{2.82}$$

The propagation equation for the true system given by Equation (2.69) with the control from Equation (2.78) is expressed as:

$$\begin{aligned}\mathbf{x}_{\mathrm{t}}(t_{i+1}) = \mathbf{\Phi}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) - \mathbf{B}_{\mathrm{d\,t}}\Big(p_1\mathbf{G}^*_{\mathrm{c}1}\hat{\mathbf{x}}_1(t_i^+) + p_2\mathbf{G}^*_{\mathrm{c}\,2}\hat{\mathbf{x}}_2(t_i^+)\ \ldots\ + p_{\mathrm{K}}\mathbf{G}^*_{\mathrm{c\,K}}\hat{\mathbf{x}}_{\mathrm{K}}(t_i^+)\Big) \\ + \mathbf{G}_{\mathrm{d\,t}}\mathbf{w}_{\mathrm{d\,t}}(t_i)\end{aligned} \tag{2.83}$$

Now substitute the filter update from Equation (2.80) into that result and expand to yield:

$$\begin{aligned}\mathbf{x}_{\mathrm{t}}(t_{i+1}) &= \mathbf{\Phi}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{\mathrm{i}}) \\ &- p_1\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c}1}\Big((\mathbf{I} - \mathbf{K}_{1*}\mathbf{H}_{1*})\hat{\mathbf{x}}_1(t_{\mathrm{i}}^-) + \mathbf{K}_1\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{\mathrm{i}}) + \mathbf{K}_1\mathbf{v}_{\mathrm{t}}(t_{\mathrm{i}})\Big) \\ &- p_2\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c}\,2}\Big((\mathbf{I} - \mathbf{K}_2\mathbf{H}_2)\hat{\mathbf{x}}_2(t_{\mathrm{i}}^-) + \mathbf{K}_2\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{\mathrm{i}}) + \mathbf{K}_2\mathbf{v}_{\mathrm{t}}(t_{\mathrm{i}})\Big)\ \ldots \\ &- p_{\mathrm{K}}\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c\,K}}\Big((\mathbf{I} - \mathbf{K}_{\mathrm{K}}\mathbf{H}_{\mathrm{K}})\hat{\mathbf{x}}_{\mathrm{K}}(t_{\mathrm{i}}^-) + \mathbf{K}_{\mathrm{K}}\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{\mathrm{i}}) + \mathbf{K}_{\mathrm{K}}\mathbf{v}_{\mathrm{t}}(t_{\mathrm{i}})\Big) \\ &+ \mathbf{G}_{\mathrm{d\,t}}\mathbf{w}_{\mathrm{d\,t}}(t_i)\end{aligned} \tag{2.84}$$

Simplification yields:

$$\begin{aligned}\mathbf{x}_{\mathrm{t}}(t_{i+1}) = (\mathbf{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{d\,t}}\sum_{\mathrm{j}=1}^{\mathrm{K}} p_{\mathrm{j}}\mathbf{G}^*_{\mathrm{c\,j}}\mathbf{K}_{\mathrm{j}}\mathbf{H}_{\mathrm{t}})\mathbf{x}_{\mathrm{t}}(t_i) - p_1\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c}1}(\mathbf{I} - \mathbf{K}_1\mathbf{H}_1)\hat{\mathbf{x}}_1(t_i^-)\ \ldots \\ - p_{\mathrm{K}}\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c\,K}}(\mathbf{I} - \mathbf{K}_{\mathrm{K}*}\mathbf{H}_{\mathrm{K}*})\hat{\mathbf{x}}_{\mathrm{K}}(t_i^-) - (\mathbf{B}_{\mathrm{d\,t}}\sum_{\mathrm{j}=1}^{\mathrm{K}} p_{\mathrm{j}}\mathbf{G}^*_{\mathrm{c\,j}}\mathbf{K}_{\mathrm{j}})\mathbf{v}_{\mathrm{t}}(t_{\mathrm{i}}) \\ + \mathbf{G}_{\mathrm{d\,t}}\mathbf{w}_{\mathrm{d\,t}}(t_i)\end{aligned} \tag{2.85}$$

Now, the augmented state equations that form the one-step prediction model for the MMAC with lower bounding use Equations (2.82) and (2.85), and is expressed as:

$$\begin{bmatrix} \mathbf{x}_{\mathrm{t}}(t_{i+1}) \\ \hat{\mathbf{x}}_1(t_{i+1}^-) \\ \vdots \\ \hat{\mathbf{x}}_{\mathrm{K}}(t_{i+1}^-) \end{bmatrix} = \begin{bmatrix} (\mathbf{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{dt}} \sum_{\mathrm{j}=1}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}_{\mathrm{cj}}^* \mathbf{K}_{\mathrm{j}} \mathbf{H}_{\mathrm{t}}) & -\mathbf{B}_{\mathrm{dt}} p_1 \mathbf{G}_{\mathrm{c1}}^* (\mathbf{I} - \mathbf{K}_1 \mathbf{H}_1) & \cdots & -\mathbf{B}_{\mathrm{dt}} p_{\mathrm{K}} \mathbf{G}_{\mathrm{cK}}^* (\mathbf{I} - \mathbf{K}_{\mathrm{K}} \mathbf{H}_{\mathrm{K}}) \\ (\mathbf{\Phi}_1 \mathbf{K}_1 \mathbf{H}_{\mathrm{t}} - \mathbf{B}_{\mathrm{d1}} \sum_{\mathrm{j}=1}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}_{\mathrm{cj}}^* \mathbf{K}_{\mathrm{j}} \mathbf{H}_{\mathrm{t}}) & \begin{bmatrix} \mathbf{\Phi}_1 (\mathbf{I} - \mathbf{K}_1 \mathbf{H}_1) \\ -\mathbf{B}_{\mathrm{d1}} p_1 \mathbf{G}_{\mathrm{c1}}^* (\mathbf{I} - \mathbf{K}_1 \mathbf{H}_1) \end{bmatrix} & \cdots & -\mathbf{B}_{\mathrm{d1}} p_{\mathrm{K}} \mathbf{G}_{\mathrm{c1}}^* (\mathbf{I} - \mathbf{K}_{\mathrm{K}} \mathbf{H}_{\mathrm{K}}) \\ \vdots & \vdots & \ddots & \vdots \\ (\mathbf{\Phi}_{\mathrm{K}} \mathbf{K}_{\mathrm{K}} \mathbf{H}_{\mathrm{t}} - \mathbf{B}_{\mathrm{dK}} \sum_{\mathrm{j}=1}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}_{\mathrm{cj}}^* \mathbf{K}_{\mathrm{j}} \mathbf{H}_{\mathrm{t}}) & -\mathbf{B}_{\mathrm{dK}} p_1 \mathbf{G}_{\mathrm{cK}}^* (\mathbf{I} - \mathbf{K}_1 \mathbf{H}_1) & \cdots & \begin{bmatrix} \mathbf{\Phi}_{\mathrm{K}} (\mathbf{I} - \mathbf{K}_{\mathrm{K}} \mathbf{H}_{\mathrm{K}}) \\ -\mathbf{B}_{\mathrm{dK}} p_{\mathrm{K}} \mathbf{G}_{\mathrm{cK}}^* (\mathbf{I} - \mathbf{K}_{\mathrm{K}} \mathbf{H}_{\mathrm{K}}) \end{bmatrix} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{\mathrm{t}}(t_i) \\ \hat{\mathbf{x}}_1(t_i^-) \\ \vdots \\ \hat{\mathbf{x}}_{\mathrm{K}}(t_i^-) \end{bmatrix}$$

$$+ \begin{bmatrix} -\mathbf{B}_{\mathrm{dt}} \sum_{\mathrm{j}=1}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}_{\mathrm{cj}}^* \mathbf{K}_{\mathrm{j}} \\ (\mathbf{\Phi}_1 \mathbf{K}_1 - \mathbf{B}_{\mathrm{d1}} \sum_{\mathrm{j}=1}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}_{\mathrm{cj}}^* \mathbf{K}_{\mathrm{j}}) \\ \vdots \\ (\mathbf{\Phi}_{\mathrm{K}} \mathbf{K}_{\mathrm{K}} - \mathbf{B}_{\mathrm{dK}} \sum_{\mathrm{j}=1}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}_{\mathrm{cj}}^* \mathbf{K}_{\mathrm{j}}) \end{bmatrix} \mathbf{v}_{\mathrm{t}}(t_i) + \begin{bmatrix} \mathbf{G}_{\mathrm{dt}} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix} \mathbf{w}_{\mathrm{dt}}(t_i) \tag{2.86}$$

As usual, this is can be expressed in the form:

$$\begin{bmatrix} \mathbf{x}_{\mathrm{t}}(t_{i+1}) \\ \hat{\mathbf{x}}_1(t_{i+1}^-) \\ \vdots \\ \hat{\mathbf{x}}_{\mathrm{K}}(t_{i+1}^-) \end{bmatrix} = \mathbf{T}_{\mathrm{LB}} \begin{bmatrix} \mathbf{x}_{\mathrm{t}}(t_i) \\ \hat{\mathbf{x}}_1(t_i^-) \\ \vdots \\ \hat{\mathbf{x}}_{\mathrm{K}}(t_i^-) \end{bmatrix} + \mathbf{L}_{\mathrm{LB}} \begin{bmatrix} \mathbf{w}_{\mathrm{dt}}(t_i) \\ \mathbf{v}_{\mathrm{t}}(t_i) \end{bmatrix} \tag{2.87}$$

where $\mathbf{T}_{\mathrm{LB}}$ and $\mathbf{L}_{\mathrm{LB}}$ are expressed as:

$$\mathbf{T}_{\mathrm{LB}}=\begin{bmatrix}(\boldsymbol{\Phi}_{\mathrm{t}}-\mathbf{B}_{\mathrm{dt}}\sum_{\mathrm{j=1}}^{\mathrm{K}}p_{\mathrm{j}}\mathbf{G}_{\mathrm{cj}}^{*}\mathbf{K}_{\mathrm{j}}\mathbf{H}_{\mathrm{t}}) & -\mathbf{B}_{\mathrm{dt}}p_{1}\mathbf{G}_{\mathrm{c1}}^{*}(\mathbf{I}-\mathbf{K}_{1}\mathbf{H}_{1}) & \cdots & -\mathbf{B}_{\mathrm{dt}}p_{\mathrm{K}}\mathbf{G}_{\mathrm{cK}}^{*}(\mathbf{I}-\mathbf{K}_{\mathrm{K}}\mathbf{H}_{\mathrm{K}})\\ (\boldsymbol{\Phi}_{1}\mathbf{K}_{1}\mathbf{H}_{\mathrm{t}}-\mathbf{B}_{\mathrm{d1}}\sum_{\mathrm{j=1}}^{\mathrm{K}}p_{\mathrm{j}}\mathbf{G}_{\mathrm{cj}}^{*}\mathbf{K}_{\mathrm{j}}\mathbf{H}_{\mathrm{t}}) & \begin{bmatrix}\boldsymbol{\Phi}_{1}(\mathbf{I}-\mathbf{K}_{1}\mathbf{H}_{1})\\-\mathbf{B}_{\mathrm{d1}}p_{1}\mathbf{G}_{\mathrm{c1}}^{*}(\mathbf{I}-\mathbf{K}_{1}\mathbf{H}_{1})\end{bmatrix} & \cdots & -\mathbf{B}_{\mathrm{d1}}p_{\mathrm{K}}\mathbf{G}_{\mathrm{c1}}^{*}(\mathbf{I}-\mathbf{K}_{\mathrm{K}}\mathbf{H}_{\mathrm{K}})\\ \vdots & \vdots & \ddots & \vdots\\ (\boldsymbol{\Phi}_{\mathrm{K}}\mathbf{K}_{\mathrm{K}}\mathbf{H}_{\mathrm{t}}-\mathbf{B}_{\mathrm{dK}}\sum_{\mathrm{j=1}}^{\mathrm{K}}p_{\mathrm{j}}\mathbf{G}_{\mathrm{cj}}^{*}\mathbf{K}_{\mathrm{j}}\mathbf{H}_{\mathrm{t}}) & -\mathbf{B}_{\mathrm{dK}}p_{1}\mathbf{G}_{\mathrm{cK}}^{*}(\mathbf{I}-\mathbf{K}_{1}\mathbf{H}_{1}) & \cdots & \begin{bmatrix}\boldsymbol{\Phi}_{\mathrm{K}}(\mathbf{I}-\mathbf{K}_{\mathrm{K}}\mathbf{H}_{\mathrm{K}})\\-\mathbf{B}_{\mathrm{dK}}p_{\mathrm{K}}\mathbf{G}_{\mathrm{cK}}^{*}(\mathbf{I}-\mathbf{K}_{\mathrm{K}}\mathbf{H}_{\mathrm{K}})\end{bmatrix}\end{bmatrix} \tag{2.88}$$

and

$$\mathbf{L}_{\mathrm{LB}}=\begin{bmatrix}\mathbf{G}_{\mathrm{dt}} & -\mathbf{B}_{\mathrm{dt}}\sum_{\mathrm{j=1}}^{\mathrm{K}}p_{\mathrm{j}}\mathbf{G}_{\mathrm{cj}}^{*}\mathbf{K}_{\mathrm{j}}\\ \mathbf{0} & (\boldsymbol{\Phi}_{1}\mathbf{K}_{1}-\mathbf{B}_{\mathrm{d1}}\sum_{\mathrm{j=1}}^{\mathrm{K}}p_{\mathrm{j}}\mathbf{G}_{\mathrm{cj}}^{*}\mathbf{K}_{\mathrm{j}})\\ \vdots & \vdots\\ \mathbf{0} & (\boldsymbol{\Phi}_{\mathrm{k}}\mathbf{K}_{\mathrm{k}}-\mathbf{B}_{\mathrm{dk}}\sum_{j=1}^{\mathrm{K}}p_{\mathrm{j}}\mathbf{G}_{\mathrm{cj}}^{*}\mathbf{K}_{\mathrm{j}})\end{bmatrix} \tag{2.89}$$

with the statistics for the noise as:

$$E\left\{\begin{bmatrix}\mathbf{w}_{\mathrm{dt}}(t_i)\\ \mathbf{v}_{\mathrm{t}}(t_i)\end{bmatrix}\begin{bmatrix}\mathbf{w}_{\mathrm{dt}}{}^{\mathrm{T}}(t_j) & \mathbf{v}_{\mathrm{t}}{}^{\mathrm{T}}(t_j)\end{bmatrix}\right\}=\begin{cases}\mathbf{Q_0}(t_i) & t_i=t_j\\ \mathbf{0} & t_i\neq t_j\end{cases} \tag{2.90}$$

and where

$$\mathbf{Q_0}(t_i)=\begin{bmatrix}\mathbf{Q}_{\mathrm{dt}}(t_i) & \mathbf{0}\\ \mathbf{0} & \mathbf{R}_{\mathrm{t}}(t_i)\end{bmatrix} \tag{2.91}$$

The output autocorrelation of the augmented system at time t_{i+1} is written as:

$$\mathrm{E}\left\{\begin{bmatrix}\mathbf{x}^{\mathrm{t}}(t_{i+1}) \\ \hat{\mathbf{x}}_1(t_{i+1}^-) \\ \vdots \\ \hat{\mathbf{x}}_{\mathrm{K}}(t_{i+1}^-)\end{bmatrix}\begin{bmatrix}\mathbf{x}_{\mathrm{t}}{}^{\mathrm{T}}(t_{i+1}) & \hat{\mathbf{x}}_1{}^{\mathrm{T}}(t_{i+1}^-) & \cdots & \hat{\mathbf{x}}_{\mathrm{K}}{}^{\mathrm{T}}(t_{i+1}^-)\end{bmatrix}\right\} = \mathbf{\Xi}_{\mathrm{k}}(t_{i+1}^-) \tag{2.92}$$

$$= \mathbf{T}_{\mathrm{LB}}\,\mathbf{\Xi}_{\mathrm{k}}(t_i^-)\,\mathbf{T}_{\mathrm{LB}}^{\mathrm{T}} + \mathbf{L}_{\mathrm{LB}}\mathbf{Q_0}\mathbf{L}_{\mathrm{LB}}^{\mathrm{T}}$$

As in the previous section, the upper the upper left quadrant of this expression is the output autocorrelation, $\mathbf{\Psi}_{\mathbf{x}_\mathrm{t}} = \mathrm{E}\{[\mathbf{x}_\mathrm{t}][\mathbf{x}_\mathrm{t}]^{\mathrm{T}}\}$ as required in Equation (2.59) for the cost evaluation of Equation (2.35). This ability to assess the cost function when there is a lower bound will be important in the sequel for determining performance when steady-state has not yet been reached and the maximum available probability has not been assumed by a single filter.

2.5.4 The Baram Distance Measure

The previous discussions centered on the assumptions that a single filter can be determined as the filter that would assume all the probability when all the filters run to steady state. Sheldon [56] used the work by Baram in order to develop a method of choosing the filter and that work is summarized here.

Recall that the hypothesis conditional probability of each of the individual filters is determined by a recursive evaluation of the computation as given in Equation (2.20). Baram [3,4,5] developed a proximity measure of the closeness of a given filter based on the conditional density that appears in the numerator of Equation (2.20) and defined in Equation (2.21). The filter that is assumed to have the maximum probability at steady state is the filter with the minimum of the proximity measure given by:

$$\ell \equiv \min \ell_{\mathrm{k}} \qquad \mathrm{k} = 1,\ldots\mathrm{K} \tag{2.93}$$

where

$$\ell_k \equiv \log|\mathbf{A}_k| + \mathrm{tr}[\mathbf{A}_k^{-1}\mathbf{N}_k] \tag{2.94}$$

This equation is derived from taking the expectation of the logarithm of the conditional density function as shown:

$$\begin{aligned}\ell_k &\equiv \mathrm{E}\left\{\log\left(\left(f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\mathbf{z}_i \mid \mathbf{a}_k, \mathbf{Z}_{i-1})\right)\right\}\\ &\equiv \mathrm{E}\left\{\log\left(\beta_k \exp\{\cdot\}\right)\right\} = \mathrm{E}\left\{\log(\beta_k)\right\} + \mathrm{E}\left\{\{\cdot\}\right\}\end{aligned} \tag{2.95}$$

where

$$\begin{aligned}\{\cdot\} &= -\tfrac{1}{2}\mathbf{r}_k{}^{\mathrm{T}}(t_i)\mathbf{A}_k^{-1}(t_i)\mathbf{r}_k(t_i)\\ &= -\tfrac{1}{2}\,\mathrm{tr}\left[\mathbf{A}_k^{-1}(t_i)\mathbf{r}_k(t_i)\mathbf{r}_k{}^{\mathrm{T}}(t_i)\right]\end{aligned} \tag{2.96}$$

and

$$\boldsymbol{\beta}_k = \frac{1}{(2\boldsymbol{\pi})^{m/2}\left|\mathbf{A}_k(t_i)\right|^{1/2}} \tag{2.97}$$

The measure of the steady state prediction error covariance of the residual computed in the kth filter is given by:

$$\mathbf{A}_k \equiv [\mathbf{H}_k][\mathbf{P}_k][\mathbf{H}_k]^{\mathrm{T}} + \mathbf{R}_k \tag{2.98}$$

$\mathbf{P}_k$ is the prediction error covariance of the kth filter and $\mathbf{H}_k$ is the kth filter measurement matrix. At steady-state, the residuals in Equation (2.96) are reflected by the steady state prediction error covariance. Thus, $\mathbf{N}_k$ in Equation (2.94) is the actual steady state prediction error autocorrelation of the kth filter and is given by:

$$\mathbf{N}_k = [\mathbf{H}_t \quad -\mathbf{H}_k]\boldsymbol{\Xi}_{k\infty}[\mathbf{H}_t \quad -\mathbf{H}_k]^{\mathrm{T}} \tag{2.99}$$

where $\boldsymbol{\Xi}_{k\infty}$ is the state prediction autocorrelation given in Equations (2.76), and $\mathbf{H}_k$ and $\mathbf{H}_t$ are the measurement matrices of the kth filter and true system, respectively. Now Equation (2.99) is used in Equation (2.96) to yield:

$$\{\cdot\} \equiv -\tfrac{1}{2}\,\mathrm{tr}\left[\mathbf{A}_k^{-1}\mathbf{N}_k\right] \tag{2.100}$$

Now substitute Equation (2.100) into Equation (2.95) to yield the proximity measure. Constant multipliers that do not change the results of the measure are not included in the final form as given in Equation (2.94).

2.6 Summary

The presentation of the MMAC basics in this chapter sets the foundation for the discussions and development in subsequent chapters. The three major subject areas are: Kalman filtering, MMAE/MMAC structures, and filter, controller and system models in the uncertain parameter space. Each area will be addressed as topics of research in subsequent chapters.

Key to the development is the Kalman filter which is elemental to all control schemes developed in this research. The subsequent discussion of the MMAE portion is based on a bank of Kalman filters, as is the development of the MMAC and MMAE-based control. Inherent in the discussions are the system equations and notation that will be used throughout the remainder of this dissertation.

Both the MMAE and MMAC depend on a set of models associated with points in an appropriate parameter space. This chapter reviews the discretization of the parameter space for optimal placement of those models. For the MMAC, the discussion and equation development was based on the standard assumption that the filter design model is the same as the controller design model. This review sets up the next chapter that lifts the filter-controller model equivalence assumption, where the equation development will follow the same approach. The final aspect of MMAC covered is the analytic determination of the closest model to the actual system as represented as a point in parameter space. In reality, without bounding the lower probability, the MMAC will converge to one filter as the closest in parameter space. This property is crucial to the discretization algorithms since the evaluation of performance depends on the selected filter-controller pair at steady state. Discretization will be vital in the development of the enhanced MMAC structures discussed in the following chapters.

Chapter 3 - A Generalized LQG Design Approach

The standard design approach for the LQG controller is based on the principle of certainty equivalence [36] in which the filter is designed separately from the full-state feedback controller. The two modules operate in tandem to yield the LQG control algorithm. It is assumed that the filter is designed based on the truth model in order to minimize a mean squared estimation error criterion. Likewise, the full-state feedback controller is designed based on the truth model to minimize a mean square regulation error criterion. Intuitively, it seems to make sense that this is the best approach for the control algorithm design. However, when one considers the design of a Luenberger observer [23] rather than a Kalman filter as the state estimator, we find that the goal is actually to speed up the dynamics of the observer relative to the truth model. This is done by choosing the observer gains to place poles of the resultant dynamics. This is not intrinsically incorporated into the solution to the Riccati equation solution to minimize the cost function when designing the corresponding Kalman filter. Thus, it seems plausible and optimal with respect to some criterion on performance of the overall LQG controller to design the Kalman filter for a model that is actually "faster" than the dynamics of the truth model.

The discussion in the following sections first develops a performance measure that incorporates the full-state feedback controller and the Kalman filter into a single cost evaluation. Minimization of this cost function will yield the optimal controller-filter combination. The minimization occurs over the space of possible controllers and filters, each based on a model possibly different from the truth model. The next section expands on the optimization algorithms. The first proposed minimization algorithm determines the best performance through the selection of parameter values of the filter and controller models in the neighborhood of the truth model parameter values. For this research, a neighborhood is defined to be composed of the parameter points that are bounded (by a specified distance) around the nominal values of the parameters of the truth model. In addition, it is assumed that the design models are not necessarily of the same dimensionality as the truth model but are based on the same uncertain parameters. Finally, with a slight modification to the minimization criterion, a second algorithm proposes to improve robustness to possible parameter variations of the system model. The perturbation about the truth model is used to specify the required robustness region.

3.1 Derivation of LQG Design Performance Measure

The discovery of improved LQG performance by using models for the controller and filter designs that are possibly different from each other and different from the truth model originated from work in studying ways to enhance the MMAC and MMAE-based control structures. The goal in that part of the research (as will be discussed in Chapters 4 and 5) is to improve performance through the selection of the best controller, given a parameter estimate. It followed naturally that the "best" controller might be different from the LQ full-state feedback controller based on the truth model, since the parameter estimate might not match exactly to the truth model. Likewise, the best filter for the given parameter estimate might also be different from the Kalman filter based on the truth model.

Derivation of the generalized LQG performance measure and the cost function to minimize parallels the derivation for the MMAC structure [35,56]. Now, that MMAC performance evaluation derivation, of course, follows the typical performance evaluation of the standard LQG stochastic regulator [35]. The first component of the cost function to minimize for the LQG design as it follows the MMAC is the expectation of the regulation output autocorrelation (mean squared regulation error) given by:

$$\begin{aligned} \mathrm{E}\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y}\} &\equiv \mathrm{E}\{\mathrm{tr}(\mathbf{W}\mathbf{y}\mathbf{y}^{\mathrm{T}})\} \\ &\equiv \mathrm{tr}(\mathbf{W}\mathrm{E}\{\mathbf{C}\mathbf{x}\mathbf{x}^{\mathrm{T}}\mathbf{C}^{\mathrm{T}}\}) \\ &\equiv \mathrm{tr}(\mathbf{W}\mathbf{C}\boldsymbol{\Psi}_{\mathbf{x}}\mathbf{C}^{\mathrm{T}}) \end{aligned} \tag{3.1}$$

where $\boldsymbol{\Psi}_{\mathbf{x}}$ =E{$\mathbf{x}\mathbf{x}^{\mathrm{T}}$}. The regulation error autocorrelation is a function of the parameter vector triple {$\mathbf{a}_{\mathrm{t}}$, $\mathbf{a}_{\mathrm{f}}$, $\mathbf{a}_{\mathrm{c}}$} where the superscripts specifies the truth model, Kalman filter model, and control model, respectively. It is assumed that these three vectors have the same dimension, but not necessarily the same values. Though the models are based on the same uncertain parameters, it is possible that the filter and control models have a reduced number of states compared to that of the truth model. For convenience in the derivation, the parameter vector **a** represents the parameters that differ in value in the respective models (taking on values $\mathbf{a}_{\mathrm{t}}$, $\mathbf{a}_{\mathrm{f}}$, and $\mathbf{a}_{\mathrm{c}}$ in the truth model, filter design model, and full-state feedback control design model, respectively).

The filter design model for the system at steady state described in terms of state $\mathbf{x}_{\mathrm{f}}$ for the parameter vector $\mathbf{a}_{\mathrm{f}}$, measurement **z**, and controlled variable **y** is given as:

$$\mathbf{x}_{\mathrm{f}}(t_{i+1}) = \boldsymbol{\Phi}_{\mathrm{f}}\mathbf{x}_{\mathrm{f}}(t_i) + \mathbf{B}_{\mathrm{df}}\mathbf{u}(t_i) + \mathbf{G}_{\mathrm{df}}\mathbf{w}_{\mathrm{df}}(t_i) \tag{3.2}$$

$$\mathbf{z}(t_i) = \mathbf{H}_{\mathrm{f}}\mathbf{x}_{\mathrm{f}}(t_i) + \mathbf{v}_{\mathrm{f}}(t_i) \tag{3.3}$$

$$\mathbf{y}(t_i) = \mathbf{C}_{\mathrm{f}}\mathbf{x}_{\mathrm{f}}(t_i) \tag{3.4}$$

with

$$E\left\{\mathbf{w}_{\mathrm{df}}(t_i) \quad \mathbf{w}_{\mathrm{df}}(t_j)^{\mathrm{T}}\right\} = \begin{cases} \mathbf{Q}_{\mathrm{df}}(t_i) & t_i = t_j \\ \mathbf{0} & t_i \neq t_j \end{cases} \tag{3.5}$$

$$\mathrm{E}\left\{\mathbf{v}_{\mathrm{f}}(t_i) \quad \mathbf{v}_{\mathrm{f}}(t_j)^{\mathrm{T}}\right\} = \begin{cases} \mathbf{R}_{\mathrm{f}}(t_i) & t_i = t_j \\ \mathbf{0} & t_i \neq t_j \end{cases} \tag{3.6}$$

$\mathbf{w}_{\mathrm{df}}(t_i)$ and $\mathbf{v}_{\mathrm{f}}(t_i)$ are assumed pair-wise independent. The control law is given by

$$\mathbf{u}(t_i) = -\mathbf{G}_{\mathrm{c}}\hat{\mathbf{x}}_{\mathrm{f}}(t_i^+) \tag{3.7}$$

where $\mathbf{G}_{\mathrm{c}}$ is the optimal control gain matrix from the solution to the deterministic LQ regulator portion of the control problem. Now $\hat{\mathbf{x}}_{\mathrm{f}}$ is the output of the Kalman filter given by

$$\begin{aligned} \hat{\mathbf{x}}_{\mathrm{f}}(t_{i+1}^-) &= \mathbf{\Phi}_{\mathrm{f}}\hat{\mathbf{x}}_{\mathrm{f}}(t_i^+) - \mathbf{B}_{\mathrm{df}}\mathbf{G}_{\mathrm{c}}\hat{\mathbf{x}}_{\mathrm{f}}(t_i^+) \\ &= \left(\mathbf{\Phi}_{\mathrm{f}} - \mathbf{B}_{\mathrm{df}}\mathbf{G}_{\mathrm{c}}\right)\hat{\mathbf{x}}_{\mathrm{f}}(t_i^+) \end{aligned} \tag{3.8}$$

and is propagated between samples, with a sampled-data measurement update given by

$$\hat{\mathbf{x}}_{\mathrm{f}}(t_i^+) = \hat{\mathbf{x}}_{\mathrm{f}}(t_i^-) + \mathbf{K}_{\mathrm{f}}[\mathbf{z}(t_i) - \mathbf{H}_{\mathrm{f}}\hat{\mathbf{x}}_{\mathrm{f}}(t_i^-)] \tag{3.9}$$

where $\mathbf{K}_{\mathrm{f}}$ is the observer gain matrix as found for the standard Kalman filter. The matrices of the Kalman filter in Equations (3.8)-(3.9) are those from the system matrices in (3.2) and (3.3). For the conventional design, it is then a matter of determining $\mathbf{K}_{\mathrm{f}}$ and $\mathbf{G}_{\mathrm{c}}$ via Riccati equations.

Consider the conventional LQG controller with a Kalman filter specified in Equations (3.8)-(3.9) based on the model $\mathbf{a}_{\mathrm{f}}$ of the system dynamics in Equations (3.2)-(3.3). Now denote the controller gain $\mathbf{G}_{\mathrm{c}}$ as $\mathbf{G}_{\mathrm{cc}}$ to indicate it is based on a controller model based on $\mathbf{a}_{\mathrm{c}}$, separate from the parameter vectors associated with the filter and truth models. So the equation for true control becomes

$$\mathbf{u}_{\mathrm{t}}(t_i) = -\mathbf{G}_{\mathrm{cc}}\hat{\mathbf{x}}_{\mathrm{f}}(t_i^+) \tag{3.10}$$

Now substitute the control into the Kalman filter propagation equation to yield:

$$\begin{aligned}\hat{\mathbf{x}}_{\mathrm{f}}(t_{i+1}^{-}) &= \mathbf{\Phi}_{\mathrm{f}}\hat{\mathbf{x}}_{\mathrm{f}}(t_i^{+}) - \mathbf{B}_{\mathrm{df}}\mathbf{G}_{\mathrm{cc}}\hat{\mathbf{x}}_{\mathrm{f}}(t_i^{+}) \\ &= \left(\mathbf{\Phi}_{\mathrm{f}} - \mathbf{B}_{\mathrm{df}}\mathbf{G}_{\mathrm{cc}}\right)\hat{\mathbf{x}}_{\mathrm{f}}(t_i^{+})\end{aligned} \tag{3.11}$$

As done in the previous development of the MMAC in Section 2.3, substitute the measurement update given in Equation (3.9) for the Kalman filter at t_i^{+}. This final substitution yields the state estimate for the filter at t_{i+1}^{-} and is given by:

$$\begin{aligned}\hat{\mathbf{x}}_{\mathrm{f}}(t_{i+1}^{-}) &= \left(\mathbf{\Phi}_{\mathrm{f}} - \mathbf{B}_{\mathrm{df}}\mathbf{G}_{\mathrm{cc}}\right)\left\{\left(\mathbf{I} - \mathbf{K}_{\mathrm{f}}\mathbf{H}_{\mathrm{f}}\right)\hat{\mathbf{x}}_{\mathrm{f}}(t_i^{-}) + \mathbf{K}_{\mathrm{f}}\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) + \mathbf{K}_{\mathrm{f}}\mathbf{v}_{\mathrm{t}}(t_i)\right\} \\ &= \left(\mathbf{\Phi}_{\mathrm{f}} - \mathbf{B}_{\mathrm{df}}\mathbf{G}_{\mathrm{cc}}\right)\left(\mathbf{I} - \mathbf{K}_{\mathrm{f}}\mathbf{H}_{\mathrm{f}}\right)\hat{\mathbf{x}}_{\mathrm{f}}(t_i^{-}) + \left(\mathbf{\Phi}_{\mathrm{f}} - \mathbf{B}_{\mathrm{df}}\mathbf{G}_{\mathrm{cc}}\right)\mathbf{K}_{\mathrm{f}}\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) \\ &\quad + \left(\mathbf{\Phi}_{\mathrm{f}} - \mathbf{B}_{\mathrm{df}}\mathbf{G}_{\mathrm{cc}}\right)\mathbf{K}_{\mathrm{f}}\mathbf{v}_{\mathrm{t}}(t_i)\end{aligned} \tag{3.12}$$

The true system is modeled by

$$\mathbf{x}_{\mathrm{t}}(t_{i+1}) = \mathbf{\Phi}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) + \mathbf{B}_{\mathrm{dt}}\mathbf{u}_{\mathrm{t}}(t_i) + \mathbf{G}_{\mathrm{dt}}\mathbf{w}_{\mathrm{dt}}(t_i) \tag{3.13}$$

Now substitute the control from Equation (3.10) and the measurement from Equation (3.9) to yield:

$$\begin{aligned}\mathbf{x}_{\mathrm{t}}(t_{i+1}) &= \mathbf{\Phi}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) - \mathbf{B}_{\mathrm{dt}}\mathbf{G}_{\mathrm{cc}}\left\{\hat{\mathbf{x}}_{\mathrm{f}}(t_i^{-}) + \mathbf{K}_{\mathrm{f}}[\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) + \mathbf{v}_{\mathrm{t}}(t_i) - \mathbf{H}_{\mathrm{f}}\hat{\mathbf{x}}_{\mathrm{f}}(t_i^{-})]\right\} \\ &\quad + \mathbf{G}_{\mathrm{dt}}\mathbf{w}_{\mathrm{dt}}(t_i) \\ &= \left(\mathbf{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{dt}}\mathbf{G}_{\mathrm{cc}}\mathbf{K}_{\mathrm{f}}\mathbf{H}_{\mathrm{t}}\right)\mathbf{x}_{\mathrm{t}}(t_i) - \mathbf{B}_{\mathrm{dt}}\mathbf{G}_{\mathrm{cc}}\left(\mathbf{I} - \mathbf{K}_{\mathrm{f}}\mathbf{H}_{\mathrm{f}}\right)\hat{\mathbf{x}}_{\mathrm{f}}(t_i^{-}) \\ &\quad - \mathbf{B}_{\mathrm{dt}}\mathbf{G}_{\mathrm{cc}}\mathbf{K}_{\mathrm{f}}\mathbf{v}_{\mathrm{t}}(t_i) + \mathbf{G}_{\mathrm{dt}}\mathbf{w}_{\mathrm{dt}}(t_i)\end{aligned} \tag{3.14}$$

Now after making all substitutions and simplifications, the augmented system description is given by

$$\begin{aligned}\begin{bmatrix}\mathbf{x}_{\mathrm{t}}(t_{i+1}) \\ \hat{\mathbf{x}}_{\mathrm{f}}(t_{i+1}^{-})\end{bmatrix} &= \begin{bmatrix}(\mathbf{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{dt}}\mathbf{G}_{\mathrm{cc}}\mathbf{K}_{\mathrm{f}}\mathbf{H}_{\mathrm{t}}) & -\mathbf{B}_{\mathrm{dt}}\mathbf{G}_{\mathrm{cc}}(\mathbf{I} - \mathbf{K}_{\mathrm{f}}\mathbf{H}_{\mathrm{f}}) \\ (\mathbf{\Phi}_{\mathrm{f}} - \mathbf{B}_{\mathrm{d}}\mathbf{G}_{\mathrm{cc}})\mathbf{K}_{\mathrm{f}}\mathbf{H}_{\mathrm{t}} & (\mathbf{\Phi}_{\mathrm{f}} - \mathbf{B}_{\mathrm{df}}\mathbf{G}_{\mathrm{cc}})(\mathbf{I} - \mathbf{K}_{\mathrm{f}}\mathbf{H}_{\mathrm{f}})\end{bmatrix}\begin{bmatrix}\mathbf{x}_{\mathrm{t}}(t_i) \\ \hat{\mathbf{x}}_{\mathrm{f}}(t_i^{-})\end{bmatrix} \\ &\quad + \begin{bmatrix}-\mathbf{B}_{\mathrm{dt}}\mathbf{G}_{\mathrm{cc}}\mathbf{K}_{\mathrm{f}} \\ (\mathbf{\Phi}_{\mathrm{f}} - \mathbf{B}_{\mathrm{df}}\mathbf{G}_{\mathrm{cc}})\mathbf{K}_{\mathrm{f}}\end{bmatrix}\mathbf{v}_{\mathrm{t}}(t_i) + \begin{bmatrix}\mathbf{G}_{\mathrm{dt}} \\ \mathbf{0}\end{bmatrix}\mathbf{w}_{\mathrm{dt}}(t_i)\end{aligned} \tag{3.15}$$

Now define

$$\mathrm{E}\left\{\begin{bmatrix}\mathbf{w}_{\mathrm{dt}}(t_i) \\ \mathbf{v}_{\mathrm{t}}(t_i)\end{bmatrix}\begin{bmatrix}\mathbf{w}_{\mathrm{dt}}(t_j)^{\mathrm{T}} & \mathbf{v}_{\mathrm{t}}(t_j)^{\mathrm{T}}\end{bmatrix}\right\} = \begin{cases}\mathbf{Q}_{\mathbf{0}}(t_i) & t_i = t_j \\ \mathbf{0} & t_i \neq t_j\end{cases} \tag{3.16}$$

where

$$\mathbf{Q_0}(t_i) = \begin{bmatrix} \mathbf{Q}_{\mathrm{dt}}(t_i) & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{\mathrm{t}}(t_i) \end{bmatrix} \tag{3.17}$$

and further define

$$\mathbf{T'} = \begin{bmatrix} (\mathbf{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{dt}}\mathbf{G}_{\mathrm{cc}}\mathbf{K}_{\mathrm{f}}\mathbf{H}_{\mathrm{t}}) & -\mathbf{B}_{\mathrm{dt}}\mathbf{G}_{\mathrm{cc}}(\mathbf{I} - \mathbf{K}_{\mathrm{f}}\mathbf{H}_{\mathrm{f}}) \\ (\mathbf{\Phi}_{\mathrm{f}} - \mathbf{B}_{\mathrm{df}}\mathbf{G}_{\mathrm{cc}})\mathbf{K}_{\mathrm{f}}\mathbf{H}_{\mathrm{t}} & (\mathbf{\Phi}_{\mathrm{f}} - \mathbf{B}_{\mathrm{df}}\mathbf{G}_{\mathrm{cc}})(\mathbf{I} - \mathbf{K}_{\mathrm{f}}\mathbf{H}_{\mathrm{f}}) \end{bmatrix} \tag{3.18}$$

$$\mathbf{L'} = \begin{bmatrix} \mathbf{G}_{\mathrm{dt}} & -\mathbf{B}_{\mathrm{dt}}\mathbf{G}_{\mathrm{cc}}\mathbf{K}_{\mathrm{f}} \\ \mathbf{0} & (\mathbf{\Phi}_{\mathrm{f}} - \mathbf{B}_{\mathrm{df}}\mathbf{G}_{\mathrm{cc}})\mathbf{K}_{\mathrm{f}} \end{bmatrix} \tag{3.19}$$

then the output autocorrelation of the augmented system expressed in Equation (3.15) can be written conveniently as:

$$E\left\{\begin{bmatrix} \mathbf{x}_{\mathrm{t}}(t_i) \\ \hat{\mathbf{x}}_{\mathrm{f}}(t_i^-) \end{bmatrix}\begin{bmatrix} \mathbf{x}_{\mathrm{t}}(t_i)^{\mathrm{T}} & \hat{\mathbf{x}}_{\mathrm{f}}(t_i^-)^{\mathrm{T}} \end{bmatrix}\right\} = \mathbf{\Xi}(t_{i+1}) = \mathbf{T'}\mathbf{\Xi}(t_i)\mathbf{T'}^{\mathrm{T}} + \mathbf{L'}\mathbf{Q_0}\mathbf{L'}^{\mathrm{T}} \tag{3.20}$$

Now the upper left partition of the resultant expectation expression is the regulation autocorrelation $\mathbf{\Psi}_{\mathbf{x}_{\mathrm{t}}}$ in Equation (3.1) for the true system. As stated previously, this development for the single filter/controller follows that of the MMAC in Chapter 2, and thus Equation (3.20) is similar to Equation (2.76).

Using only the output correlation (mean squared regulation error) as the performance measure for the design of the best controller would be consistent with a *cheap control* version of LQG control, i.e. one in which the quadratic cost on states strongly dominates the quadratic on control. This was defined in Chapter 2 as:

$$\mathrm{J}_{2\,\mathrm{c}} \equiv \frac{\int_A E\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y}\}da}{\int_A da} \tag{3.21}$$

More generally, the design of the best controller is based on the sum of the output correlation with a quadratic on control consistent with the cost used to define the LQG controller in the first place. Thus, the more general form of cost function expressed in Equation (3.21) is:

$$J'_{2\,c} \equiv \frac{\int_A \mathrm{E}\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y} + \mathbf{u}^{\mathrm{T}}\mathbf{U}\mathbf{u}\}da}{\int_A da} \tag{3.22}$$

where **u** is the control and **U** is the weight used to allow different emphasis on individual control components.

The quadratic on the control is derived similarly to the quadratic on the states and is denoted as

$$\begin{aligned} \mathrm{E}\{\mathbf{u}^T\mathbf{U}\mathbf{u}\} &\equiv \mathrm{E}\{\mathrm{tr}(\mathbf{U}\mathbf{u}\mathbf{u}^{\mathrm{T}})\} \\ &\equiv \mathrm{tr}(\mathbf{U}\mathrm{E}\{\mathbf{G}_{cc}\hat{\mathbf{x}}\hat{\mathbf{x}}^{\mathrm{T}}\mathbf{G}_{cc}{}^{\mathrm{T}}\}) \\ &\equiv \mathrm{tr}(\mathbf{U}\mathbf{G}_{cc}\boldsymbol{\Psi}_{\hat{x}}\mathbf{G}_{cc}{}^{\mathrm{T}}) \end{aligned} \tag{3.23}$$

Here $\boldsymbol{\Psi}_{\hat{x}}$ is the lower right partition output autocorrelation of the augmented system expressed in Equation (3.20). Now put together the position correlation and the quadratic on control to yield the resultant cost function:

$$\mathrm{E}\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y}\} + \mathrm{E}\{\mathbf{u}^{\mathrm{T}}\mathbf{U}\mathbf{u}\} \equiv \mathrm{tr}(\mathbf{W}\mathbf{C}\boldsymbol{\Psi}_{x}\mathbf{C}^{\mathrm{T}}) + \mathrm{tr}(\mathbf{U}\mathbf{G}_{cc}\boldsymbol{\Psi}_{\hat{x}}\mathbf{G}_{cc}{}^{\mathrm{T}}) \tag{3.24}$$

The form of this cost function is the same as the cost function used to design the control gain matrix $\mathbf{G}_{cc}$. However, the minimization of this cost function is dependent on both the designed Kalman filter and the control gain matrix. Thus, the cost is minimized by searching over the available models that yield the components $\boldsymbol{\Psi}_x$ and $\boldsymbol{\Psi}_{\hat{x}}$

3.2 Modified LQG Design Algorithm

The previous section developed the position correlation equation for the case in which the controller, filter and truth models are possibly different. In Equations (3.9) and (3.10), and thus also in Equations (3.15) and (3.20), the controller and filter gains denoted $\mathbf{K}_f$ and $\mathbf{G}_{cc}$ are selected, based on solutions to two Riccati equations which are based on assumed models not necessarily equivalent to the truth model. The object is to determine the gains $\mathbf{K}_f$ and $\mathbf{G}_{cc}$ which minimize the position correlation (mean squared values of regulation errors) when the resulting modified LQG (MLQG) controller is used in a real-world environment described by a particular truth model. Again, the conventional LQG approach assumes that $\mathbf{K}_f$ and $\mathbf{G}_{cc}$ are determined based on models equivalent to the truth model (or reduced-order version thereof, but still based on the same parameter value $\mathbf{a}_t$). The approaches for the MLQG discussed in the following subsections remove the assumptions that the controller and filter design models must match the assumed truth model.

3.2.1 Modified LQG With Optimally Selected Filter Parameter $\mathbf{a}_f$

Consider the assumption that the actual system states are completely and perfectly measurable. The problem of controller design then becomes the classical LQR approach. Now of course, the *best* filter would need to be added if the states were not perfectly measurable. As previously noted in the discussion on the Luenberger observer, the indicated solution to the filter/controller gain search is to determine the full-state feedback controller and then find the optimal filter by selecting the parameter $\mathbf{a}_f$ from the range of the parameter space. Based on this assessment, the following is the filter/controller selection algorithm for the modified LQG control algorithm:

1. For the given system truth model, design a controller using typical LQR techniques to obtain the gain $\mathbf{G}_{cc}$ (i.e., assume $\mathbf{a}_c = \mathbf{a}_t$; further note that $\mathbf{a}_c = \mathbf{a}_t$

implies, for an online adaptive system to be considered in the subsequent chapters, that $\mathbf{a}_c$ is equal to $\hat{\mathbf{a}}$).

2. Select a representative filter parameter $\mathbf{a}_f$ in the neighborhood of the true system parameter $\mathbf{a}_t$ and design a Kalman filter based upon $\mathbf{a}_f$, using the typical LQE techniques but based on $\mathbf{a}_f$ rather than $\mathbf{a}_t$ (i.e., assume $\mathbf{a}_f \neq \mathbf{a}_t$ generally).
3. Compute the position correlation using $\mathbf{G}_{cc}$ and the Kalman Filter from steps (1) and (2), respectively.
4. Choose a vector minimization procedure to find the optimal filter parameter $\mathbf{a}_f$ that minimizes the position correlation from step (3).

The minimization as described in step (4) finds the minimal position correlation as a function of the filter model parameter vector. However, since this could involve many applications of solving Riccati equations to accomplish step (2), it might be computationally more feasible to check discrete points across the defined parameter space for the filter model. This brute force method would only yield an approximate solution, or it could be used to define a better starting point for a minimization algorithm.

In the previous discussion of Luenberger observers, it was assumed that the model that corresponds to the Kalman filter would be *faster* than the actual system model. *Faster* is used to convey that the model corresponds to a filter that has shortened the response time of the system. Hence, following this line of reasoning, the result of the minimization in step (4) should yield a filter that either matches the actual system model or it is *faster*.

3.2.2 Modified LQG with Optimally Selected Controller Parameter $\mathbf{a}_c$

Rather than designing the controller to match the actual system as done in the previous subsection, this section proposes to design the filter to match the system model (i.e., $\mathbf{a}_f = \mathbf{a}_t$) and then find the *best* controller. This *best* controller is designed using a model based on the parameter $\mathbf{a}_c$ optimally selected from the range of the parameter space. Designing the filter to match the actual system may seem contrary to the previous assumptions based on Luenberger observers. However, if the controller gain were increased, this has a similar effect as speeding up the filter in relation to the actual system model.

Based on the assumption that the filter parameter is fixed, the following is the filter/controller design algorithm for the modified LQG controller with optimally selected controller parameter:

1. For the given system truth model, design a Kalman filter (i.e., similar to the previous section, but for the filter, assume $\mathbf{a}_f = \mathbf{a}_t$; further note that $\mathbf{a}_f = \mathbf{a}_t$ implies, for an online adaptive system to be considered in the subsequent chapters, that $\mathbf{a}_f$ is equal to $\hat{\mathbf{a}}$).
2. Select a representative controller model parameter vector $\mathbf{a}_c$ that is in the neighborhood of the system truth model parameter $\mathbf{a}_t$ and design the controller using the typical LQR techniques to find G_{cc} based on that $\mathbf{a}_c$ (i.e. assume $\mathbf{a}_c \neq \mathbf{a}_t$ generally).
3. Compute the position correlation using the **K**alman filter and $\mathbf{G}_{cc}$ from steps (1) and (2).

4. Choose a vector minimization procedure to find the optimal controller parameter $\mathbf{a}_c$ that minimizes the position correlation from step (3).

In the previous section it was assumed that the model that corresponds to the Kalman filter will be faster than the actual system model. For the control, the dynamics of the system are sped up by a larger controller gain. A larger gain will correspond to a controller design model that is slower than the actual system model. This is logical: if large excursions from the desired state or output values are assumed to *persist* longer in time, it is beneficial to use larger gains to drive them more strongly towards the desired values. Hence, an application of this design algorithm should demonstrate that, with the filter designed for the actual system model, the controller model should either match the actual system model or it should be *slower*. In other words, the controller model will have a larger gain than what would be ordinarily associated with the actual system model.

3.2.3 Generalized Approach to the Modified LQG

The approach in this subsection combines the concepts of the previous two subsections and allows the filter model and the controller model both to differ from the assumed system truth model. Rather than fixing the controller model or the filter model to match the actual system model, the optimization algorithm shall determine both of these models which will yield the best performance.

The optimization algorithm that implements the generalized modified LQG controller is specified as follows:

1. Select a representative filter model parameter ($\mathbf{a}_f \neq \mathbf{a}_t$ generally, but in the neighborhood of $\mathbf{a}_t$) and design the corresponding Kalman filter.

2. Select a representative controller model parameter ($\mathbf{a}_c \neq \mathbf{a}_t$ generally, but in the neighborhood of $\mathbf{a}_t$) and design a full-state feedback controller using the typical LQR techniques to find G_{cc}.
3. Compute the position correlation using Kalman filter and $\mathbf{G}_{cc}$ from steps (1) and (2).
4. Choose a vector minimization procedure to find the filter parameter $\mathbf{a}_f$ and the controller parameter $\mathbf{a}_c$ that minimizes the position correlation from step (3).

In the previous two subsections, the effects of the filter and the controller were considered individually. First, it was assumed that the model that corresponds to the Kalman filter would be *faster* than the actual system model or the controller design model. For the control, the dynamics of the system is made faster by a larger gain. A larger gain will correspond to a controller model that is slower than the actual system model. For the generalized MLQG, the effects of the filter and controllers must be considered together. Hence, it is not really possible to predict how each model will be selected in relationship to the system model. For example, the gain for the control could be selected so large, that the dynamics of the filter do not have to be faster than, or even as fast as, the system model. However, it is anticipated that the filter based on the parameter $\mathbf{a}_f$ will always correspond to a faster model than controller model based on the parameter $\mathbf{a}_c$.

3.2.4 Tradeoff among Modified LQG Approaches

It is assumed that the three approaches discussed in this section will yield different performances that most likely will be dependent on the system parameters. The tradeoff will be the performance in terms of output correlation versus the amount of control that is required to implement the controller scheme, which can be predicted by:

$$\begin{aligned} \mathrm{E}\{\mathbf{u}^{\mathrm{T}}\mathbf{W}\mathbf{u}\} &= \mathrm{tr}\left(\mathbf{W}\mathrm{E}\{\mathbf{u}\mathbf{u}^{\mathrm{T}}\}\right) \\ &= \mathrm{tr}\left(\mathbf{W}\mathbf{G}_{\mathrm{cc}}\mathbf{\Psi}_{\hat{\mathrm{x}}}\mathbf{G}_{\mathrm{cc}}{}^{\mathrm{T}}\right) \end{aligned} \tag{3.25}$$

where $\mathbf{W}$ is an appropriately chosen weighting matrix and $\mathbf{\Psi}_{\hat{\mathrm{x}}}$ is the lower right partition output autocorrelation of the augmented system expressed in Equation (3.20). For the design of a real world system, it is often the saturation of the controller that will limit the amount of control. It will then be up to the designer to determine the outcome of the tradeoff.

3.3 LQG Selection Algorithm Modification for Robust Controller

The design methods for the modified LQG controller discussed in the previous section assumed that the truth model was based on a parameter vector $\mathbf{a}_{\mathrm{t}}$ that does not vary. In addition, the truth model is assumed accurate (i.e., that $\mathbf{a}_{\mathrm{t}}$ is known perfectly). Now consider the case (most likely to occur in an actual system) that the assumed truth model may not be exactly the same as the actual system model. For simplification, assume that the difference between the actual system model and the defined truth model can be captured in the specification of the parameter vector $\mathbf{a}_{\mathrm{t}}$. In terms of statistics, assume that the variation of a scalar parameter is some scalar multiple of the standard deviation given as:

$$\mathrm{a}_{\mathrm{t}} \in [\mathrm{a}_{\mathrm{nominal}} - k\sigma,\ \mathrm{a}_{\mathrm{nominal}} + k\sigma] \tag{3.26}$$

for some chosen scalar k, where σ is the standard deviation. Correspondingly for a *vector* parameter, assume $\mathbf{a}_{\mathrm{t}}$ is within the ellipsoid centered at $\mathbf{a}_{\mathrm{nominal}}$ and defined by the eigenvectors and scalar k times the square root of each of the eigenvalues of the covariance $\mathbf{P}_{\mathbf{aa}}$ of **a** values. As k is made larger, as going from 0 to 1 to 2, etc., larger and larger sets of possible $\mathbf{a}_t$ values are allowed, and thus greater and greater amounts of robustness is provided if the controller is designed to perform acceptably against this entire set of possible $\mathbf{a}_t$ values. This variation over the possible parameter space is illustrated in Figure 3.1.

The goal for the selection algorithm is to determine the controller/filter combination for the given value of $a_{nominal}$ and σ (or $\mathbf{a}_{nominal}$ and $\mathbf{P}_{\mathbf{aa}}$ in the vector case) that can be considered robust across the whole range of possible $\mathbf{a}_t$. The algorithm to accomplish this is:

1. Select a value for the controller model parameter location in the range of possible $\mathbf{a}_t$ values.
2. For the controller selected in (1), determine the filter model that yields the minimum position correlation over the range.
3. The filter selected is determined by first computing the maximum position correlation over the admissible range of $\mathbf{a}_t$ values for each possible filter. Of those maximum position correlation values, choose the filter corresponding to the

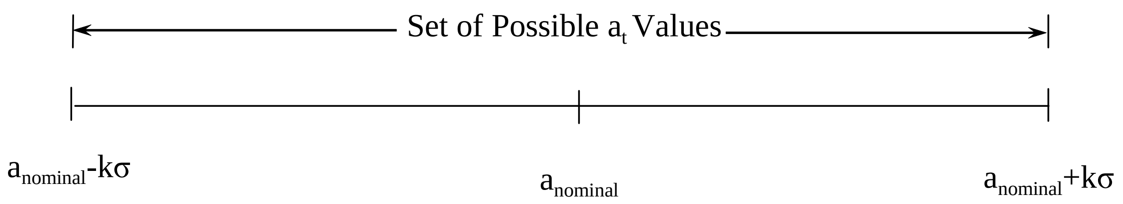

Figure 3.1 Range over which a_t can occur, given a particular $a_{nominal}$ and σ

minimum of those computed maxima. These first 3 steps are illustrated in Figure 3.2a and given as:

$$F \equiv F_i \text{ which yields } \min(\max \boldsymbol{\Psi}_{C_1,F_i})$$

A second approach could use the RMS of $\boldsymbol{\Psi}$ over the possible range of $\mathbf{a}_t$ values for each possible filter:

$$F \equiv F_i \text{ which yields } \min(\text{RMS } \boldsymbol{\Psi}_{C_1,F_i})$$

4) Filter selection process is repeated for each controller in the parameter space bounded by Equation (3.26)

5) Similar to the filter selection, the controller is determined by using the position correlation over the admissible range of $\mathbf{a}_t$ for the controller and filter combination from step (4). Of those maximum position correlation values,

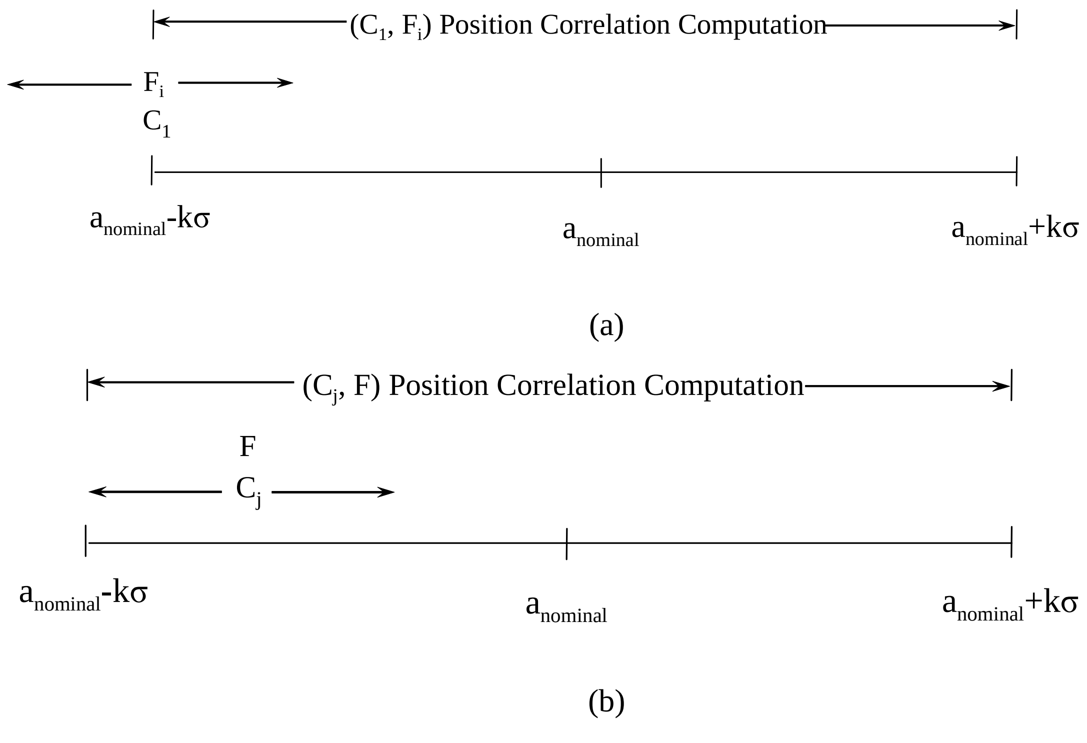

Figure 3.2(a) Filter search given a controller in the parameter space (b) Controller search and filter found for each possible controller

choose the controller and filter corresponding to the minimum of those computed maxima. These two steps (4) and (5) are illustrated in Figure 3.2b and given as:

$$C \equiv C_j \text{ which yields } \min(\max \boldsymbol{\Psi}_{C_j,F_i})$$

As done with the filter selection, a second approach could use the RMS of $\boldsymbol{\Psi}$ over the possible range of $\mathbf{a}_t$ values for the controller and filter combination from step (4)

$$C \equiv C_j \text{ which yields } \min(\text{RMS}\, \boldsymbol{\Psi}_{C_j,F_i})$$

6) Steps (1) through (5) are repeated for each possible value of $\mathbf{a}_t$ in the parameter space.

3.4 Summary

This chapter develops several modifications to the typical LQG design approach. The primary change removes the standard assumption that the Kalman filter design model is the same as that used for the controller. As with the typical methods and under assumed certainty equivalence, the Kalman filter is designed and combined with a controller developed with LQR methods, both based on an assumed system model. To evaluate the effectiveness of the filter/controller combination, a performance measure based on the output error autocorrelation is developed. This performance measure is a generalization of the MMAC evaluation equations with just one filter/controller combination rather than bank of multiple LQG controllers. Also, the filter and the controller models for that single combination are specified independently rather than as an LQG-like controller. Now, the optimization for best performance in terms of minimized regulation error becomes a search over the possible filters and controllers. Hence, this development sets the stage for application of the modified LQG design to the MMAC. The second modification to the typical LQG design builds upon the first and addresses robustness to variation of the possible true system. Rather than assuming a constant system model in the performance evaluation, it is now assumed that the system can exist in a known range. Either the RMS value or maximum value of the performance evaluation over the possible system values can be used to determine the optimal filter controller combination for the end LQG design. This generalized LQG design approach also has application to MMAC.

The two design approaches and the work underlying them actually came from the initial investigation of this research into improved MMAC design. Hence, the application to MMAC will be investigated in the next chapter.

Chapter 4 - MMAC Development

This chapter discusses the research accomplished to advance the design and evaluation of the MMAC architecture. In his dissertation research, Sheldon [56,57] made significant contributions to MMAC design through optimization of the discretization of filter models in the space of uncertain parameters. Sheldon was not concerned so much with the performance characteristics of the individual components of the MMAC, but with, given their characteristics, how should the assumed parameter values of these components be placed in parameter space. The original intention of the current research was to develop and demonstrate improvements to the MMAE-based control by improving the properties of the single full-state feedback controller in that architecture. However, it became obvious that any improvements to a single controller within MMAE-based control should also apply to the blended controller elements in the MMAC architecture. Thus, the insights that came from the modified LQG design are used in the controller elements of the MMAC architecture to improve the overall performance characteristics.

The enhancements to the elemental controllers result in modifications to the MMAC design algorithms. This chapter first presents improvements to the MMAC design synthesis developed by Sheldon through a revision to the evaluation step in the discretization algorithm. Next, this chapter develops the replacement of the typical LQG controllers in the MMAC with the modified LQG controllers of the previous chapter. This discussion includes the modified LQG controller with enhanced robustness in the MMAC architecture. The next section develops an approach that replaces the control gain matrix in the conventional MMAC architecture with a full-state feedback control element to produce a generalized MMAC (GMMAC). The final section presents an evaluation tool that determines the variance of the selection of the filter in the MMAC at steady state. Though this tool is not directly used in the optimal placement of the filters in parameter space, it can be used to predict performance of the MMAC.

4.1 Modification to the MMAC Evaluation and Discretization

As reviewed in Chapter 2, the evaluation of the MMAC is a two-step process in which the MMAE equations are evaluated, followed by an evaluation of the MMAC equations. The MMAE equations evaluation determines which filter is closest in probability at steady state and the MMAC equations determine the position error autocorrelation. However, review of the derivation of the position error autocorrelation gives insight into a necessary modification of the MMAC evaluation and associated discretization algorithm that has proven useful for all MMACs, and particularly for enhanced performance of the modified MMAC and GMMAC algorithms in the subsequent sections.

The minimization of the cost function for the MMAC requires evaluations of a weighted position autocorrelation as the truth model varies over the parameter space of concern. The cost function evaluation uses weighted state autocorrelations, as is shown in the following:

$$J_{2c} \equiv \frac{\int_A \mathrm{E}\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y}\}da}{\int_A da} \tag{4.1}$$

where $\mathbf{y} = \mathbf{C}\mathbf{x}$ and

$$\mathrm{E}\left\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y}\right\} = \mathrm{tr}\left(\mathbf{W}\mathbf{C}_{\mathrm{t}}\,\boldsymbol{\Psi}_{\mathbf{x}_{\mathrm{t}}}\mathbf{C}_{\mathrm{t}}^{\mathrm{T}}\right) \tag{4.2}$$

$$\boldsymbol{\Psi}_{\mathbf{x}_{\mathrm{t}}} = \mathrm{E}\left\{\mathbf{x}_{\mathrm{t}}\,\mathbf{x}_{\mathrm{t}}^{\mathrm{T}}\right\} \tag{4.3}$$

and **W** is the weighting matrix used for emphasizing the importance of individual outputs relative to the other outputs, and $\mathbf{C}_t$ is the system output matrix. Though Equation (4.1) uses only the output, the true states to compute that output are dependent on the states of the selected filter denoted as *sel* from k = 1…K filter/controllers. The autocorrelation of the true states, $\mathbf{x}_t$ as well as the filter state estimates, $\hat{\mathbf{x}}_{sel}$ or more generally $\hat{\mathbf{x}}_k$, are determined by solving the Lyponuv equation for the MMAC denoted in Chapter 2 as:

$$\mathrm{E}\left\{\begin{bmatrix}\mathbf{x}_t(t_{i+1}) \\ \hat{\mathbf{x}}_k(t_{i+1}^-)\end{bmatrix}\begin{bmatrix}\mathbf{x}_t^T(t_{i+1}) & \hat{\mathbf{x}}_k^T(t_{i+1}^-)\end{bmatrix}\right\} = \mathbf{\Xi}_k(t_{i+1}^-) = \mathbf{T}\,\mathbf{\Xi}_k(t_i^-)\,\mathbf{T}^T + \mathbf{L}\mathbf{Q_0}\mathbf{L}^T \tag{4.4}$$

where

$$\mathbf{T} = \begin{bmatrix}(\mathbf{\Phi}_t - \mathbf{B}_{dt}\mathbf{G}_{ck}^*\mathbf{K}_k\mathbf{H}_t) & -\mathbf{B}_{dt}\mathbf{G}_{ck}^*(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k) \\ (\mathbf{\Phi}_k - \mathbf{B}_{dk}\mathbf{G}_{ck}^*)\mathbf{K}_k\mathbf{H}_t & (\mathbf{\Phi}_k - \mathbf{B}_{dk}\mathbf{G}_{ck}^*)(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\end{bmatrix} \tag{4.5}$$

and

$$\mathbf{L} = \begin{bmatrix}\mathbf{G}_{dt} & -\mathbf{B}_{dt}\mathbf{G}_{ck}^*\mathbf{K}_k \\ \mathbf{0} & (\mathbf{\Phi}_k - \mathbf{B}_{dk}\mathbf{G}_{ck}^*)\mathbf{K}_k\end{bmatrix} \tag{4.6}$$

Also, note that the statistics for the modeled noise are given as:

$$\mathrm{E}\left\{\begin{bmatrix}\mathbf{w}_{dt}(t_i) \\ \mathbf{v}_t(t_i)\end{bmatrix}\begin{bmatrix}\mathbf{w}_{dt}(t_j)^T & \mathbf{v}_t(t_j)^T\end{bmatrix}\right\} = \begin{cases}\mathbf{Q_0}(t_i) & t_i = t_j \\ \mathbf{0} & t_i \neq t_j\end{cases} \tag{4.7}$$

$$\mathbf{Q_0}(t_i) = \begin{bmatrix}\mathbf{Q}_{dt}(t_i) & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_t(t_i)\end{bmatrix} \tag{4.8}$$

$\mathbf{\Psi}_{\mathbf{x}_t}$ is the upper left partition of the resultant steady-state value $\mathbf{\Xi}_{k\,\infty}$ for the selected filter *sel*. These equations reflect that the MMAC is a full-state feedback closed loop system. Observe that the input control matrix $\mathbf{B}_d$ ($\mathbf{B}_{dk}$ and $\mathbf{B}_{dt}$) and control gain $\mathbf{G}_c^*$ appear in terms for both the true and the kth or selected filter state autocorrelation.

Now consider the MMAE equations that are used in the determination of the closest filters in the sense that it is the filter with the maximum probability. For each possible filter in the bank (kth filter for k = 1…K), the autocorrelation of the true states, $\mathbf{x}_t$ as well as the filter state estimates, $\hat{\mathbf{x}}_k$ are determined by solving the Lyponuv equation for the MMAE denoted in Chapter 2 as:

$$\mathrm{E}\left\{\begin{bmatrix}\mathbf{x}_t(t_{i+1})\\ \hat{\mathbf{x}}_k(t_{i+1}^-)\end{bmatrix}\begin{bmatrix}\mathbf{x}_t^{\mathrm{T}}(t_{i+1}) & \hat{\mathbf{x}}_k^{\mathrm{T}}(t_{i+1}^-)\end{bmatrix}\right\} = \mathbf{\Xi}_k(t_{i+1}^-) = \mathbf{Y}\mathbf{\Xi}_k(t_i^-)\mathbf{Y}^{\mathrm{T}} + \mathbf{G_0Q_0G_0}^{\mathrm{T}} \tag{4.9}$$

where

$$\mathbf{Y} = \begin{bmatrix}\mathbf{\Phi}_t & \mathbf{0}\\ \mathbf{\Phi}_k\mathbf{K}_k\mathbf{H}_t & \mathbf{\Phi}_k(\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\end{bmatrix} \tag{4.10}$$

and

$$\mathbf{G_0} = \begin{bmatrix}\mathbf{0} & \mathbf{G}_{d\,t}\\ \mathbf{\Phi}_k\mathbf{K}_k & \mathbf{0}\end{bmatrix} \tag{4.11}$$

Note that the noise statistics are the same as in Equations (4.7)-(4.8).

The MMAE Equations (4.9)-(4.11) lack the control-input terms, $\mathbf{B}_d$ ($\mathbf{B}_{d\,k}$ and $\mathbf{B}_{d\,t}$) and the control gain matrix $\mathbf{G}_c^*$, since these equations are based on an open-loop system with no feedback of the control input to the MME bank of filters. An actual implementation of the MMAC uses the feedback of the control input to the MME bank of filters. Hence, the lower right partition of $\mathbf{\Xi}_{k\infty}$ is the actual solution for the state autocorrelation in the kth filter. The state estimate autocorrelation of the kth filter as given by the MMAE equations do not have the control feedback terms. Remember that the MMAE equations determine which filter is selected in steady state in the original MMAC performance evaluation algorithm developed by Sheldon. Now it has to be determined if

$\boldsymbol{\Xi}_{\infty}$, but more specifically, if the steady-state autocorrelation of the filters (the appropriate *partition* of $\boldsymbol{\Xi}_{\infty}$) computed from the MMAC equations can be used to determine the filter in the MME portion of the MMAC with the maximum probability (which may not be one if there is lower bounding placed on the filters) at steady state.

Because of the feedback control's effect on the state estimation, the control input affects the computation of the probability weighting for each filter/controller combination in an actual implementation of an MMAC. The prediction tool also must take into account the effect of the feedback control, which the proximity measure developed by Baram [3,4,5] does. This measure is used to determine the filter model closest to the true system model in the sense that the corresponding filter probability is maximized. As reviewed in Section 2.5.3, the filter that is assumed to have the maximum probability at steady state is the filter with the minimum of the proximity measure given by:

$$\ell \equiv \min \ell_{\mathrm{k}} \qquad \mathrm{k} = 1, \ldots \mathrm{K} \tag{4.12}$$

where

$$\ell_{\mathrm{k}} \equiv \log |\mathbf{A}_{\mathrm{k}}| + \mathrm{tr}[\mathbf{A}_{\mathrm{k}}^{-1} \mathbf{N}_{\mathrm{k}}] \tag{4.13}$$

$\mathbf{A}_{\mathrm{k}\infty}$ is the covariance of the steady state residuals in the kth filter, i.e., $\left[\mathbf{H}_{\mathrm{k}}\, \mathbf{P}_{\mathrm{k}\infty}\, \mathbf{H}_{\mathrm{k}}^{\mathrm{T}} + \mathbf{R}_{\mathrm{k}}\right]$. This error covariance is different from the actual steady state autocorrelation of the estimation errors in the kth filter given by:

$$\mathbf{N}_{\mathrm{k}\infty} = [-\mathbf{H}_{\mathrm{t}} \quad \mathbf{H}_{\mathrm{k}}] \boldsymbol{\Xi}_{\mathrm{k}\infty} [-\mathbf{H}_{\mathrm{t}} \quad \mathbf{H}_{\mathrm{k}}]^{\mathrm{T}} \tag{4.14}$$

where $\boldsymbol{\Xi}_{\mathrm{k}\infty}$ is the state prediction autocorrelation computed by either Equation (4.4) or Equation (4.9), and $\mathbf{H}_{\mathrm{k}}$ and $\mathbf{H}_{\mathrm{t}}$ are the output matrix of the kth filter and truth model respectively. Thus, the MMAC evaluation reduces to only computing the closed loop steady-state prediction autocorrelation using the MMAC equations, which is a simplification of the previous evaluation process.

Since the method for choosing a filter/controller in steady state is the only change, the majority of Sheldon's MMAC design procedure remains the same. This is a change to step 4 of the 5-step design procedure to approximate and to minimize the appropriate cost function numerically [56,57]. The change is summarized as follows:

4) Use a numerical integration technique to evaluate Equation (4.1).

   a) Compute $\mathbf{\Xi}_{k\infty}$ using Equation (4.4) at discrete points in the parameter space (for $\mathbf{a}_t$ value) for each filter: k = 1,…K.

   b) At each discrete point evaluate the proximity measure using $\mathbf{\Xi}_{k\infty}$ for k = 1,…K to determine the convergence to a single elemental filter/controller with the maximum probability at steady state. Denote that selected filter/controller as *sel*.

   c) For the selected filter/controller, determine $\mathbf{\Psi}_{\mathbf{x}_t\,sel}$ from the lower right partition of $\mathbf{\Xi}_{sel\infty}$ saved from the previous evaluation of $\mathbf{\Xi}_{k\infty}$ for k = 1,…K and use in the evaluation of the cost function.

Like the original MMAC discretization optimization algorithm, the five-step procedure is accomplished off-line, and the parameter values corresponding to the optimally placed models are stored for use in real time. The major difference of this algorithm is that only $\mathbf{\Xi}_{sel\,\infty}$ is computed once using the MMAC Equations (4.4)-(4.6) rather than also evaluating the MMAE equations for filter selection. This computational advantage should speed up the optimization by nearly 50 percent. More importantly, using the MMAC equations for the filter selection more closely represents the real world implementation of the MMAC. For an actual implementation, the computed control is fed back to the filters.

4.2 The Modified MMAC

As was discussed in Chapter 3, the development of the modified LQG designs followed from the MMAE-based control structure research. In the MMAC control structure, the MMAE portion serves not only to determine the probability weighting on the control, but the state estimation as well. The elemental controller is exactly an LQG controller designed at a specific point in parameter space. The filter in the MMAE and the control gain matrix define the LQG controller. Since the LQG controller is an inherent component of the MMAC, any enhancements can be incorporated into the MMAC.

The previous chapter presents an approach to improve the design of the LQG point controllers in which the design model for the filter is possibly different than the controller design model. This section incorporates this modified LQG design approach to improve point designs in the MMAC. As will be discussed, this modification will have a minor effect on the discretization algorithms. Also, since the modified MMAC only affects the procedures for the design of the components of the MMAC, the actual architecture of the MMAC will not change. The development begins with the performance evaluation equations used in the discretization algorithms.

4.2.1 Modified MMAC Performance Equations Development

The development of the performance equations for the modified MMAC follows that of the typical MMAC. As such, it is required to develop an expression for the output autocorrelation that will be used in the evaluation of the cost function given in Equation (4.1). What will be slightly different in this development is that the equations will have to take into account the fact that the models upon which the filters and controllers are designed will not necessarily be the same. This is the significant aspect developed in Chapter 3.

For the MMAC, it is necessary to presume that the filter-assumed parameter value is fixed in parameter space and that the controller can be selected from a set of controllers. Thus, of the three modified LQG controller designs discussed in Section 3.2, the modified LQG with a selected controller gain must be used as the elemental controller. This is required since the discretization for the MMAC places the filters in parameter space as opposed to the controllers. As is the case for the filter, it is assumed that the controllers are designed based on models in the neighborhood of the actual system model. Thus, the performance evaluations will be in terms of fixed filters and controllers that are selected.

The development begins with the specification of the model for the filter in the MMAE. If the filter is based on a model with the parameter $\mathbf{a}_{\mathrm{k}}$, then the state is given by:

$$\mathbf{x}_{\mathrm{k}}(t_{i+1}) = \mathbf{\Phi}_{\mathrm{k}}(t_{i+1},t_i)\mathbf{x}_{\mathrm{k}}(t_i) + \mathbf{B}_{\mathrm{d\,k}}(t_i)\mathbf{u}_{\mathrm{k}}(t_i) + \mathbf{G}_{\mathrm{d\,k}}(t_i)\mathbf{w}_{\mathrm{d\,k}}(t_i) \tag{4.15}$$

However, as discussed, the corresponding controller gain is designed based on a controller model separate from the filter and truth models. This controller gain vector is specified by $\mathbf{G}^*_{\mathrm{ck'}}$ where the k´ denotes this difference in models. So now the equation for control becomes:

$$\mathbf{u}_{\mathrm{k}}(t_i) = -\mathbf{G}^*_{\mathrm{c\,k'}}\hat{\mathbf{x}}_{\mathrm{k}}(t_i^+) \tag{4.16}$$

Now it is assumed that the MMAC has converged to a single filter/controller combination and that the control is given by the gain corresponding to the selected filter.

Hence, the control for the truth model corresponds to the control of the k$^{\text{th}}$ filter selected by convergence in probability to 1. So now the control for the true system is given by:

$$p_{sel} = 1 \quad \Rightarrow \quad \mathbf{u}_{\mathrm{t}}(t_i) = \mathbf{u}_{\mathrm{k}}(t_i) \tag{4.17}$$

and by substitution, the expression is given by

$$\mathbf{u}_{\mathrm{t}}(t_i) = -\mathbf{G}^*_{\mathrm{ck}'}\hat{\mathbf{x}}_{\mathrm{k}}(t_i^+) \tag{4.18}$$

This form of control is identical to the controller in the development of the MMAC reviewed in Chapter 2. The modification simply changes the terms corresponding to the controller gain. Thus, the development will follow the MMAC and, without further derivation, the state equations are given as (recall Equation (2.67)):

$$\begin{aligned}\begin{bmatrix}\mathbf{x}_{\mathrm{t}}(t_{i+1})\\ \hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^-)\end{bmatrix} &= \begin{bmatrix}(\boldsymbol{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{dt}}\mathbf{G}^*_{\mathrm{ck}'}\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}}) & -\mathbf{B}_{\mathrm{dt}}\mathbf{G}^*_{\mathrm{ck}'}(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\\ (\boldsymbol{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}'})\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}} & (\boldsymbol{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}'})(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\end{bmatrix}\begin{bmatrix}\mathbf{x}_{\mathrm{t}}(t_i)\\ \hat{\mathbf{x}}_{\mathrm{k}}(t_i^-)\end{bmatrix}\\ &+ \begin{bmatrix}-\mathbf{B}_{\mathrm{dt}}\mathbf{G}^*_{\mathrm{ck}'}\mathbf{K}_{\mathrm{k}}\\ (\boldsymbol{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}'})\mathbf{K}_{\mathrm{k}}\end{bmatrix}\mathbf{v}_{\mathrm{t}}(t_i) + \begin{bmatrix}\mathbf{G}_{\mathrm{dt}}\\ \mathbf{0}\end{bmatrix}\mathbf{w}_{\mathrm{dt}}(t_i)\end{aligned} \tag{4.19}$$

As in Chapter 2, define

$$\mathbf{T}_{\mathrm{Mod}} = \begin{bmatrix}(\boldsymbol{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{dt}}\mathbf{G}^*_{\mathrm{ck}'}\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}}) & -\mathbf{B}_{\mathrm{dt}}\mathbf{G}^*_{\mathrm{ck}'}(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\\ (\boldsymbol{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}'})\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}} & (\boldsymbol{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}'})(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\end{bmatrix} \tag{4.20}$$

and

$$\mathbf{L}_{\mathrm{Mod}} = \begin{bmatrix}\mathbf{G}_{\mathrm{dt}} & -\mathbf{B}_{\mathrm{dt}}\mathbf{G}^*_{\mathrm{ck}'}\mathbf{K}_{\mathrm{k}}\\ \mathbf{0} & (\boldsymbol{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}^*_{\mathrm{ck}'})\mathbf{K}_{\mathrm{k}}\end{bmatrix} \tag{4.21}$$

The output autocorrelation of the augmented system express in Equation (4.19) now is written conveniently as:

$$E\left\{\begin{bmatrix}\mathbf{x}_{\mathrm{t}}(t_{i+1})\\ \hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^-)\end{bmatrix}\begin{bmatrix}\mathbf{x}_{\mathrm{t}}{}^{\mathrm{T}}(t_{i+1}) & \hat{\mathbf{x}}_{\mathrm{k}}{}^{\mathrm{T}}(t_{i+1}^-)\end{bmatrix}\right\} = \boldsymbol{\Xi}_{\mathrm{k}}(t_{i+1}^-) = \mathbf{T}_{\mathrm{Mod}}\boldsymbol{\Xi}_{\mathrm{k}}(t_i^-)\mathbf{T}_{\mathrm{Mod}}^{\mathrm{T}} + \mathbf{L}_{\mathrm{Mod}}\mathbf{Q}_{\mathbf{0}}\mathbf{L}_{\mathrm{Mod}}^{\mathrm{T}} \tag{4.22}$$

Define the upper left quadrant of $\boldsymbol{\Xi}_{k\infty}$ as $\boldsymbol{\Psi}_{\mathbf{x}_t} = \mathrm{E}\{\mathbf{x}_t\,\mathbf{x}_t^{T}\}$ which is the expression necessary for Equation (4.2) to be used in the evaluation the cost function in Equation (4.1). Of course, $\boldsymbol{\Xi}_{k\infty}$ in Equation (4.22) will be used in the expression in Equation (4.14), which is part of the Baram measure calculation to determine the filter that has absorbed all the probability at steady state. That determination will be necessary for the discretization algorithm discussed in the next section.

4.2.2 Modified MMAC Discretization

The discretization algorithm for the modified MMAC follows the general form as discussed in Section 4.1. What is implicit in the conventional MMAC discretization algorithm is that the control gain matrix and Kalman filter of the inherent LQG control structure are available for the evaluation of the output autocorrelation. This assumes that the LQG controller designs for each parameter ($\mathbf{a}_1 \ldots \mathbf{a}_K$) have been accomplished. Of course, the typical MMAC will use the conventional LQG design in specifying the Kalman filter and the controller gain. This step in the discretization will change for the modified MMAC. Hence, rather than using the standard LQG design approach, the modfied LQG design from Chapter 3 is used. As specified in this section, the filter is fixed to the parameter and the best controller gain is determined from a model-assumed parameter value in parameter space.

4.3 The Modified MMAC with Enhanced Robustness

The modified MMAC with enhanced robustness is an approach to add robustness to the MMAC through design of the LQG controller elements. One may argue that, if the MMAC is adaptable, then the there is not a need for robustness. However, the MMAC is adaptable for the specified parameters, but not unmodeled parameters. Thus, there is a potential advantage for the elemental LQG controller to be robust to parameters not in the parameter space specified for the MMAC. A second point to consider and the subject of this section, is that at steady state, the adaptability is limited to the number of LQG controllers that form the MMAC. Thus, though the MMAC is adaptable, there is the potential for enhanced robustness to the limitations of adaptability.

Consider that at steady state, there is in effect only one LQG controller. That controller is of course selected because it is composed of the filter with the highest probability, which is closest (in the Baram distance measure sense; see the next subsection) in the specified parameter space to the true system. In all instances except those in which the filter model happens to match the truth model, the LQG controller will not be the best design for the true system. Additionally, if the true system parameter changes, that will not affect the selected filter at steady state until the parameters have changed significantly enough such that another filter is closer in probability. Thus, the LQG has to be robust enough to account for the changes in the system, even for the parameters for which it is meant to adapt.

The development of the modified MMAC with enhanced robustness will follow the development in the previous section. Since the addition of the robustness is an extension of the modified MMAC, much of the development will be presented without further derivation. Before the design approach is developed, the next section defines the region in the parameter space over which the controller should be robust.

4.3.1 Robustness Defined by the Baram Boundary

The amount of robustness necessary for the LQG controller element can be addressed in part by the amount of change in the system parameter (defined over the parameter space) until a different filter absorbs all probability. At steady state, the filter that will attain the available probability is determined by the Baram distance measure [3,4,5,56] computed for each filter as expressed in Equation (4.12). The computation of the individual distance measures in Equation (4.13) is, in fact, a function of the assumed true parameter, $\mathbf{a}_t$. Of course, as the true parameter varies over the parameter space, there is a point where the filter with the minimum Baram distance measure transitions to the next filter. Figure 4.1 is an example of the transition point in the one-dimensional case. This transition between the two filters is defined as the *Baram boundary*. When at steady state and a specific LQG controller has been selected, the change in the system parameter basically has to cross the Baram boundary before an adjacent LQG controller is selected. Thus, a certain degree of robustness of control over the region defined by the Baram boundary has the potential to improve performance over that region.

Evaluation of the Baram boundary is a matter of computing the Baram distance measures over the parameter space and noting the transition of minimums between filters. This of course requires enough evaluations of Equation (4.13) for each parameter dimension in order to obtain a refined mapping. For the one-dimensional case shown in Figure 4.1, the Baram boundary can be easily found by a search over the parameter space using the bisection method. The one-dimensional case also yields easily defined boundaries that can serve as the region to define the robustness of the controller.

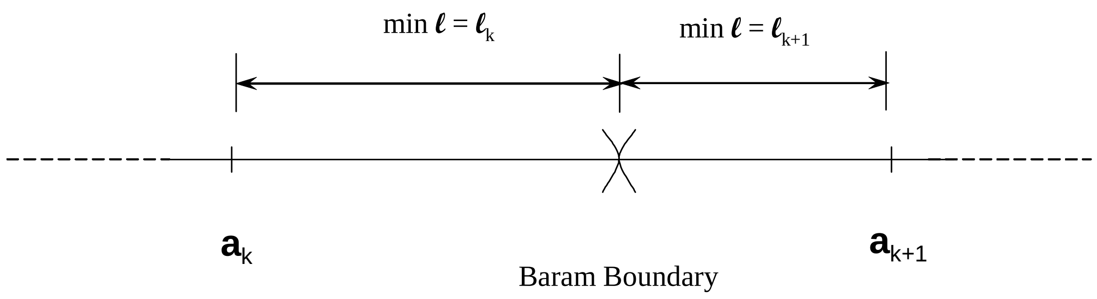

Figure 4.1 Baram boundary between two filters in parameter space

However, multi-dimensional parameter spaces will most likely yield more complex boundaries than does the one-dimensional case.

There is of course a tradeoff for the robustness. Define a measure of robustness as the RMS value of the performance evaluations over a given region of possible true system parameters. Performance at some points will suffer in order to improve the overall RMS performance across the designated portion of the parameter space. For application of the robust LQG techniques, the question to be solved in the sequel is how to determine how robust the controller should be.

4.3.2 Performance Evaluation of the Modified MMAC with Enhanced Robustness

As noted in the introduction, the modified MMAC with enhanced robustness is a simple extension to the modified MMAC discussed in the previous section. The modification only involves the specification of the controller gain matrix. Filters in the MME portion of the MMAC still provide the state estimates. As in the previous section, it is assumed that the controller is designed using a model that may not be the same as the filter model or the truth model. Thus, the development begins as in the previous section with the controller gain specified by $\mathbf{G}^*_{ck'}$ where the k´ denotes this difference in models. The control is specified as in Equation (4.18). It follows that Equations (4.19) through (4.22) are the same for the modified MMAC with enhanced robustness.

In the previous chapter, to design a LQG controller with enhanced robustness, it was assumed that the filter model was also different from the true system model as well as the controller model. That detail is also captured in Equations (4.19) through (4.22). This is different from saying the filters that determine the probabilities for blending are different from the filters that form the state estimates. An approach that uses models for the filters that determine the controllers that are also different from the filter models that form the state estimates for control is the subject of Section 4.4.

4.3.3 Discretization of Modified MMAC with Enhanced Robustness

Discretization for the conventional MMAC places a specified number of LQG controllers designed for a single point such that the cost function given by Equation (4.1) is minimized. Adding robustness to the control elements means that the LQG controllers are not designed for a single point, but for a region about the point. In the design approach to LQG with enhanced robustness from Chapter 3, the goal was to create a *radius of robustness* about a point in parameter space. To apply this to the MMAC, the regions of robustness are placed in parameter space to return the minimal cost. Now, it is a matter of how to determine the regions.

Recall that, in Chapter 3, the design of the LQG controller with robustness was for an assumed truth model specified by single point in parameter space. The modification to the LQG design incorporated a search for both a filter and a controller that would minimize the output correlation evaluated over a region of deviation from the assumed truth model (defined by the radius of robustness). Similarly, for the conventional MMAC discretization scheme, the filters are placed to minimize the output correlation over the entire assumed parameter space. The result is that each filter/controller combination provides the best performance for the portion of the parameter space specified by the Baram boundary. However, those controllers are designed for a single *point* in parameter space, whereas the robust LQG filter/controller combinations will be designed for a specified *region*.

For the application of robust LQG, as shown in Figure 4.2, the filter/controller combination has a designed amount of robustness. Since the hypothesis' conditional probability is computed based on the filters in the MME, the discretization must use $\mathbf{a}_{f_k}$ as the filter-assumed value to place in parameter space. Thus, the design of the robust LQG controller will use the filter specified by $\mathbf{a}_{f_k}$ and find the best controller gain matrix specified by $\mathbf{a}_{c_k}$, for the region defined by the robustness radius about $\mathbf{a}_{f_k}$. In this approach, the design point is the same as the filter location rather than finding a filter that might be different from the design point. This is a slight modification to the original LQG description with enhanced robustness, as outlined in Chapter 3. Only the full-state

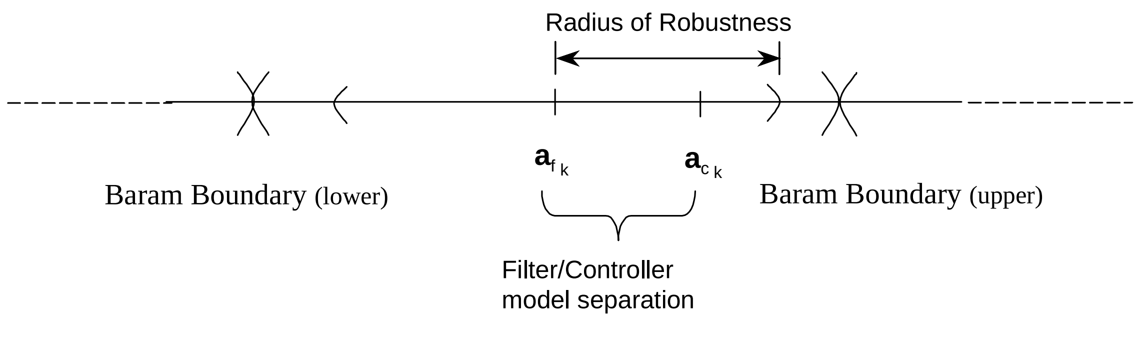

Figure 4.2 Placement of filter/controller for robustness

feedback controller must be determined for the desired radius of robustness rather than designing both the filter and the full-state feedback controller.

Discretization for the modified MMAC with enhanced robustness requires two actions to occur simultaneously, the placement of the filters and the design of the control gain matrices associated with the filters. The control associated with the filters will affect their placement and, of course, the associated Baram boundary. However, if the goal is to provide robustness over the entire region defined by the Baram boundaries, then the Baram boundaries have to be determined prior to the design of the control. But, coverage over the entire region defined by the Baram boundaries may not provide the minimal cost. Thus, the approach taken in this research is to specify the filter locations and the associated region of robustness. The cost function in Equation (4.1) now becomes dependent on not only the filter location but also the region of robustness about the filter location and is now expressed as:

$$J_{2\mathrm{c}}(\mathbf{r}) \equiv \frac{\int_A \mathrm{E}\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y}\}da}{\int_A da} \tag{4.23}$$

where **r** is the radius of robustness. Equation (4.23) now takes into account the radius of robustness that is inherent in the modified LQG design that is not expressed in cost function in Equation (4.1).

Incorporating the changes to account for the change in the evaluation of the cost function of Equation (4.23) and the design of the robust LQG controllers slightly modifies the discretization algorithm from Section 4.1. This modified algorithm for MMAC with enhanced robustness is given as:

1) Describe in terms of the parameter vector **a**, the truth model of the system, the filter and the controller, and the radius of robustness **r**, for each filter controller combination

2) Choose the number of filters, K, in the MMAE.

3) Choose a representative parameter set, $\mathcal{A} \equiv \{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_K\}$ and an initial set of radii of robustness $\mathcal{R} \equiv \{\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_K\}$, to begin the minimization.

4) Use a numerical integration technique to evaluate Equation $J_{2c}(\mathbf{r})$.

   a) Design the controller for the parameter set $\mathcal{A}$ and the corresponding robustness of radius $\mathcal{R}$.

   b) Compute $\Xi_{k\infty}$ using Equation (4.22) at discrete points in the parameter space (for $\mathbf{a}_t$ value) for each filter: k = 1,…K.

   c) At each discrete point evaluate the proximity measure using $\Xi_{k\infty}$ for k = 1,…K to determine the convergence to a single elemental filter/controller with the maximum probability at steady state. Denote that selected filter/controller as *sel*.

   d) For the selected filter/controller, determine $\boldsymbol{\Psi}_{\mathbf{x}_t\, sel}$ from the upper left partition of $\Xi_{sel\,\infty}$ saved from the previous evaluation of $\Xi_{k\infty}$ for k = 1,…K and use in the evaluation of the cost function.

5) Use a vector minimization technique with the functional evaluation from Step (4) to minimize $J_{2c}(\mathbf{r})$.

4.4 A Generalized MMAC Approach

In the previous sections, the modified MMAC design approach removes the conventional restriction that the controller models for the regulator design are the same as the corresponding filter models. As a further extension to the modified MMAC, the approach outlined in this section proposes to replace the regulator gain ($\mathbf{G}_c^*$ assumed to multiply an input $\hat{\mathbf{x}}$) with a modified LQG controller (assumed to have a measurement input, $\mathbf{z}$) based on the design method from Chapter 3. This proposed architecture illustrated in Figure 4.3 separates the design of the components into a Parallel Filter Bank (PFB) design and control element design. Thus, for this implementation, there are three design models: PFB design models, controller Kalman filter design models, and the controller gain design models. As in the typical MMAC structure, the conditional

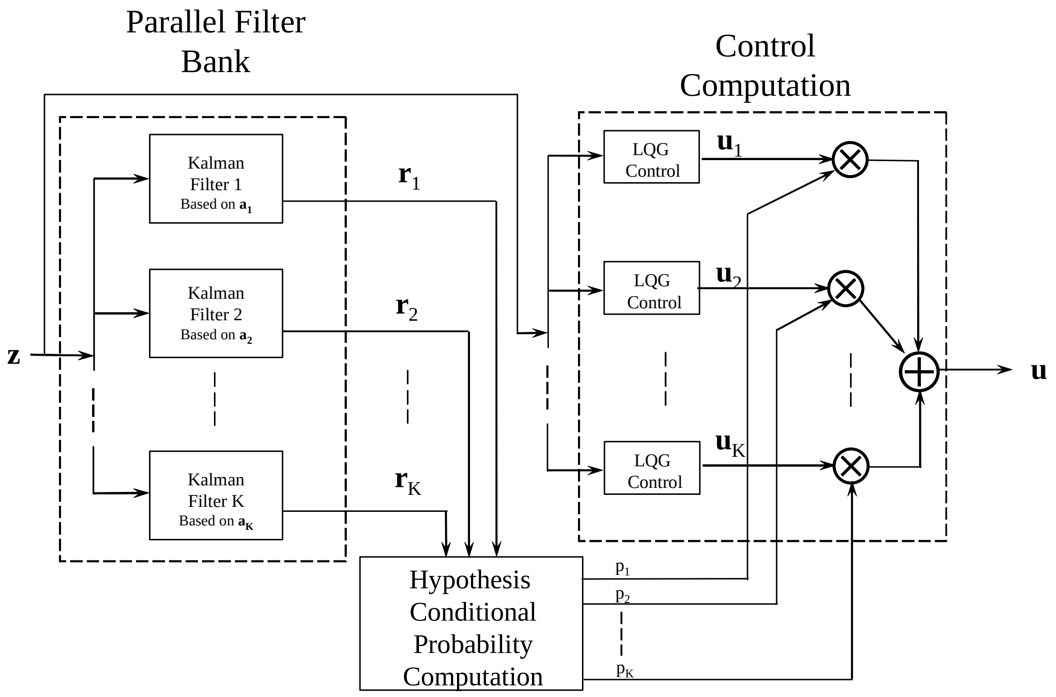

Figure 4.3 Modified MMAC structure using LQG as the control elements

probability computation is used to assign the relative weightings of the elemental control components. Thus, the elemental control components are tied to the PFB by the probability weighting.

Some of the main characteristics of the MMAC algorithm still apply to the generalized MMAC approach. First, control is derived from a blending of the individual LQG components. This blending occurs until the PFB has settled to the point at which a single filter has assumed all the probability, which is the second similarity. The Hypothesis Conditional Probability Computation converges to a single filter with a probability of one (or some defined upper bound, as covered in Section 4.4.4). Finally, though there are now three models for the design, the two models for the control element are directly tied to a corresponding model in the PFB by the probability weighting. Though the three models are different, they form a model triple consisting of the Kalman filter gains of the PFB and the Kalman filter gain within the LQG controllers and the gain matrix of the controllers, respectively, and designated by ($\mathbf{K}_f$, $\mathbf{K}_{f*}$, $\mathbf{G}_{c*}^{*}$)

The advantage of the GMMAC is that it can mitigate the tradeoff between discretization for optimal control and for optimal parameter estimation performance. The disadvantage of the typical MMAC is that the filters are used in the elemental LQG controllers and must be discretized for optimal control. As reviewed in Chapter 2, this discretization is not the same as is used for optimized parameter estimation. Similarly for the GMMAC, since the individual LQG controllers are logically tied to the filters in the PFB, there will be one discretization of those filters that the will provide the optimal control. However, the elemental LQG controllers are only logically tied to the filters in the PFB and do not rely on the state estimates from the PFB filters to form the control.

Thus, though the filters in the PFB are discretized for some other criterion such as optimal parameter estimation, the individual LQG controllers still are designed for optimal control.

As discussed in the previous section, at steady state a single filter in the PFB will have the maximum available probability for the region of parameter space as determined by the Baram bounds. The corresponding LQG controller is designed to be optimal over that region of parameter space. Further, the aggregation of optimal control over the ranges associated with the filters does not necessarily yield optimal control over the entire range of the parameters. To illustrate the effects of PFB filter placement in parameter space, Figure 4.4 shows the performance for two different discretizations of

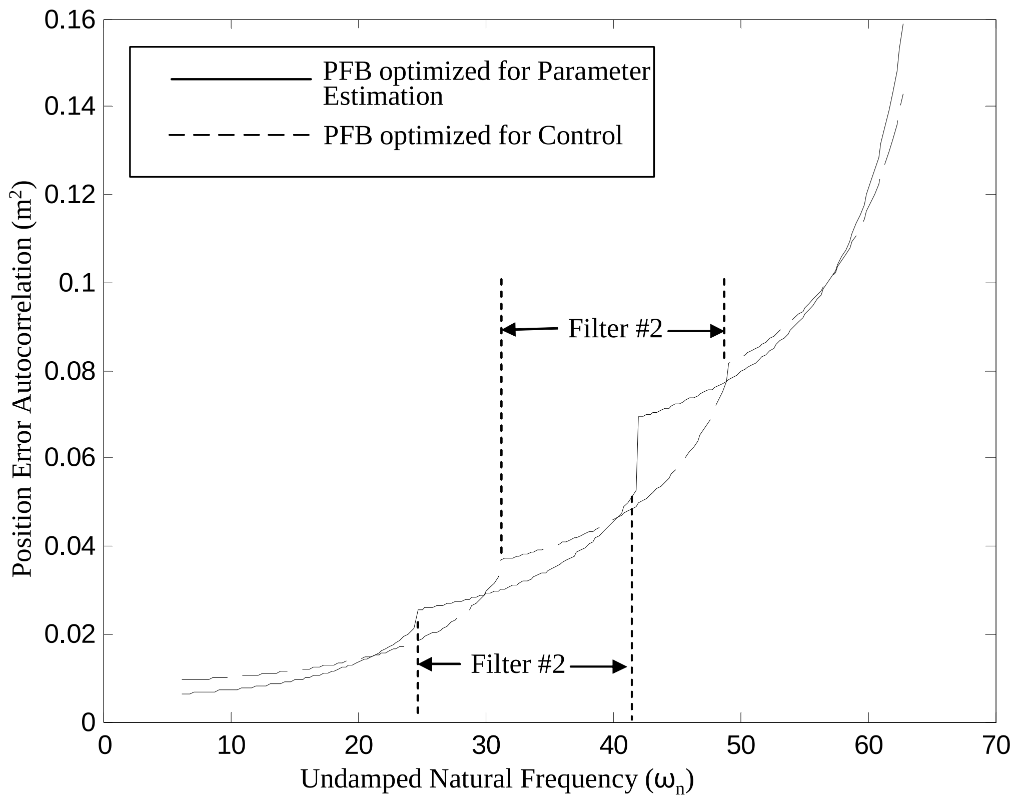

Figure 4.4 Comparison of two different PFB discretizations for the generalized MMAC

the PFB. Indicated on the plots is the performance of the second filter over the range of parameter space for each implementation. The difference in the ranges is a result of the discretization. Though each controller is optimal over the corresponding parameter space, the performance is clearly different over the common regions. The optimization also has to take into account the regions that are not in common. Thus, the tradeoff between optimal control and optimal parameter estimation still exists, but the versatility of the GMMAC at least allows for optimal control over the subsets of the ranges of the parameter space. In comparison to the MMAC, it is expected that there will be some points where the GMMAC is not lowest curve when performance is compared point-for-point, but the integral cost of the performance over the entire parameter space will be the smallest.

4.4.1 Generalized MMAC Performance Evaluation Equation Development

As with the previous developments, the first step in design of the generalized MMAC is to develop the performance evaluation equation. The goal is to determine the steady state output autocorrelation that will be used in Equation (4.1). Unlike the previous developments, there will be additional states from the full-state feedback controller that will be a part of the evaluation. Thus, the state equations to consider are the PFB filter models, the truth model and the models used in the state estimators of the controllers.

First consider the state equations associated with the filters in the PFB of the GMMAC. The propagation equation of the k$^{\text{th}}$ Kalman filter in the bank is based on the model given by the parameter vector $\mathbf{a}_{\mathrm{k}}$ and is given by:

$$\hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^{-}) = \mathbf{\Phi}_{\mathrm{k}}\hat{\mathbf{x}}_{\mathrm{k}}(t_i^{+}) + \mathbf{B}_{\mathrm{d\,k}}\mathbf{u}_{\mathrm{k}*}(t_i) \tag{4.24}$$

The constant-gain full-state feedback controllers are implemented in the form of:

$$\mathbf{u}_{k*}(t_i) = -\mathbf{G}^*_{ck*}\hat{\mathbf{x}}_{k*}(t_i^+) \tag{4.25}$$

where the propagation equation of the Kalman filter for the control based on the model given by the parameter vector $\mathbf{a}_{k*}$ is given by:

$$\hat{\mathbf{x}}_{k*}(t_{i+1}^-) = \mathbf{\Phi}_{k*}\hat{\mathbf{x}}_{k*}(t_i^+) + \mathbf{B}_{dk*}\mathbf{u}_{k*}(t_i) \tag{4.26}$$

The update equation for the Kalman filter in the FPB is given by:

$$\hat{\mathbf{x}}_k(t_i^+) = \hat{\mathbf{x}}_k(t_i^-) + \mathbf{K}_k[\mathbf{z}_i - \mathbf{H}_k(t_i)\hat{\mathbf{x}}_k(t_i^-)] \tag{4.27}$$

and the update for the Kalman filter in the control element is:

$$\hat{\mathbf{x}}_{k*}(t_i^+) = \hat{\mathbf{x}}_{k*}(t_i^-) + \mathbf{K}_{k*}[\mathbf{z}_i - \mathbf{H}_{k*}\hat{\mathbf{x}}_{k*}(t_i^-)] \tag{4.28}$$

For both Kalman filters, the measurement is given by:

$$\mathbf{z}_i = \mathbf{H}_t\mathbf{x}_t(t_i) + \mathbf{v}_t(t_i) \tag{4.29}$$

Now substitute Equation (4.29) into Equation (4.27) to yield:

$$\hat{\mathbf{x}}_k(t_i^+) = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\hat{\mathbf{x}}_k(t_i^-) + \mathbf{K}_k\mathbf{H}_t\mathbf{x}_t(t_i) + \mathbf{K}_k\mathbf{v}_t(t_i) \tag{4.30}$$

Likewise substitute Equation (4.29) into Equation (4.28) to yield:

$$\hat{\mathbf{x}}_{k*}(t_i^+) = (\mathbf{I} - \mathbf{K}_{k*}\mathbf{H}_{k*})\hat{\mathbf{x}}_{k*}(t_i^-) + \mathbf{K}_{k*}\mathbf{H}_t\mathbf{x}_t(t_i) + \mathbf{K}_{k*}\mathbf{v}_t(t_i) \tag{4.31}$$

Equations (4.30) and (4.31) are the expressions for the update equations for the PFB filters and the state estimates in the controller that are necessary for the final form of the state equation developed next.

First, for the filter propagation equations, substitute Equation (4.25) into Equation (4.24) to yield:

$$\hat{\mathbf{x}}_k(t_{i+1}^-) = \mathbf{\Phi}_k\hat{\mathbf{x}}_k(t_i^+) - \mathbf{B}_{dk*}\mathbf{G}^*_{ck*}\hat{\mathbf{x}}_{k*}(t_i^+) \tag{4.32}$$

Now substitute Equations (4.30) and (4.31) into Equation (4.32) to yield:

$$\begin{aligned}\hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^{-}) = \mathbf{\Phi}_{\mathrm{k}}\Big((\mathbf{I}-\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\hat{\mathbf{x}}_{\mathrm{k}}(t_{\mathrm{i}}^{-}) + \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{\mathrm{i}}) + \mathbf{K}_{\mathrm{k}}\mathbf{v}_{\mathrm{t}}(t_{\mathrm{i}})\Big) \\ -\mathbf{B}_{\mathrm{dk}}\mathbf{G}_{\mathrm{ck*}}^{*}\Big((\mathbf{I}-\mathbf{K}_{\mathrm{k*}}\mathbf{H}_{\mathrm{k*}})\hat{\mathbf{x}}_{\mathrm{k*}}(t_{\mathrm{i}}^{-}) + \mathbf{K}_{\mathrm{k*}}\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{\mathrm{i}}) + \mathbf{K}_{\mathrm{k*}}\mathbf{v}_{\mathrm{t}}(t_{\mathrm{i}})\Big)\end{aligned} \tag{4.33}$$

Combining terms and further simplification yields:

$$\begin{aligned}\hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^{-}) = \mathbf{\Phi}_{\mathrm{k}}(\mathbf{I}-\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\hat{\mathbf{x}}_{\mathrm{k}}(t_{i}^{-}) + (\mathbf{\Phi}_{\mathrm{k}}\mathbf{K}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}_{\mathrm{ck*}}^{*}\mathbf{K}_{\mathrm{k*}})\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{i}) \\ -\mathbf{B}_{\mathrm{dk}}\mathbf{G}_{\mathrm{ck*}}^{*}(\mathbf{I}-\mathbf{K}_{\mathrm{k*}}\mathbf{H}_{\mathrm{k*}})\hat{\mathbf{x}}_{\mathrm{k*}}(t_{i}^{-}) + (\mathbf{\Phi}_{\mathrm{k}}\mathbf{K}_{\mathrm{k}} - \mathbf{B}_{\mathrm{dk}}\mathbf{G}_{\mathrm{ck*}}^{*}\mathbf{K}_{\mathrm{k*}})\mathbf{v}_{\mathrm{t}}(t_{i})\end{aligned} \tag{4.34}$$

Equation (4.34) demonstrates that, since the controller output is fed to the PFB, the filters are dependent on the true system and the control state estimates.

Next, the truth model equations are derived in a similar procedure as was done for the MMAC in Chapter 2. The propagation equation for the true system with the control from Equation (4.25) substituted into the expression is given by:

$$\mathbf{x}_{\mathrm{t}}(t_{i+1}) = \mathbf{\Phi}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{i}) - \mathbf{B}_{\mathrm{dt}}\mathbf{G}_{\mathrm{ck*}}^{*}\hat{\mathbf{x}}_{\mathrm{k*}}(t_{i}^{+}) + \mathbf{G}_{\mathrm{dt}}\mathbf{w}_{\mathrm{dt}}(t_{i}) \tag{4.35}$$

Now substitute Equation (4.31) into Equation (4.35) to yield

$$\begin{aligned}\mathbf{x}_{\mathrm{t}}(t_{i+1}) = \mathbf{\Phi}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{i}) - \mathbf{B}_{\mathrm{dt}}\mathbf{G}_{\mathrm{ck*}}^{*}\Big((\mathbf{I}-\mathbf{K}_{\mathrm{k*}}\mathbf{H}_{\mathrm{k*}})\hat{\mathbf{x}}_{\mathrm{k*}}(t_{\mathrm{i}}^{-}) + \mathbf{K}_{\mathrm{k*}}\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{\mathrm{i}}) + \mathbf{K}_{\mathrm{k*}}\mathbf{v}_{\mathrm{t}}(t_{\mathrm{i}})\Big) \\ +\mathbf{G}_{\mathrm{dt}}\mathbf{w}_{\mathrm{dt}}(t_{i})\end{aligned} \tag{4.36}$$

Combining like terms and further simplification gives:

$$\begin{aligned}\mathbf{x}_{\mathrm{t}}(t_{i+1}) = (\mathbf{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{dt}}\mathbf{G}_{\mathrm{ck*}}^{*}\mathbf{K}_{\mathrm{k*}}\mathbf{H}_{\mathrm{t}})\mathbf{x}_{\mathrm{t}}(t_{i}) - \mathbf{B}_{\mathrm{dt}}\mathbf{G}_{\mathrm{ck*}}^{*}(\mathbf{I}-\mathbf{K}_{\mathrm{k*}}\mathbf{H}_{\mathrm{k*}})\hat{\mathbf{x}}_{\mathrm{k*}}(t_{i}^{-}) \\ -\mathbf{B}_{\mathrm{dt}}\mathbf{G}_{\mathrm{ck*}}^{*}\mathbf{K}_{\mathrm{k*}}\mathbf{v}_{\mathrm{t}}(t_{i}) + \mathbf{G}_{\mathrm{dt}}\mathbf{w}_{\mathrm{dt}}(t_{i})\end{aligned} \tag{4.37}$$

Clearly, since the control is fed to the true system, the state equations from the controller appear in Equation (4.37). However, what is noticeably missing are the state equations from the filter. This, of course, is because the control is not explicitly derived from the filter in the PFB.

Now, since the controller elements are separate from the PFB elements in the GMMAC, the state description for the Kalman filters of the controller must be included in the evaluation. The state estimation propagation equation for the controller, with the control from Equation (4.25) substituted into the expression, is given by

$$\begin{aligned}\hat{\mathbf{x}}_{k*}(t_{i+1}^-) &= \mathbf{\Phi}_{k*}\mathbf{x}_{k*}(t_i^+) - \mathbf{B}_{dk*}\mathbf{G}^*_{ck*}\hat{\mathbf{x}}_{k*}(t_i^+) \\ &= \left(\mathbf{\Phi}_{k*} - \mathbf{B}_{dk*}\mathbf{G}^*_{ck*}\right)\hat{\mathbf{x}}_{k*}(t_i^+)\end{aligned} \tag{4.38}$$

Now substitute Equation (4.31) into Equation (4.38) to yield

$$\hat{\mathbf{x}}_{k*}(t_{i+1}^-) = \left(\mathbf{\Phi}_{k*} - \mathbf{B}_{dk*}\mathbf{G}^*_{ck*}\right)\left((\mathbf{I} - \mathbf{K}_{k*}\mathbf{H}_{k*})\hat{\mathbf{x}}_{k*}(t_i^-) + \mathbf{K}_{k*}\mathbf{H}_t\mathbf{x}_t(t_i) + \mathbf{K}_{k*}\mathbf{v}_t(t_i)\right) \tag{4.39}$$

Grouping like terms yields:

$$\begin{aligned}\hat{\mathbf{x}}_{k*}(t_{i+1}^-) = &\left(\mathbf{\Phi}_{k*} - \mathbf{B}_{dk*}\mathbf{G}^*_{ck*}\right)\mathbf{K}_{k*}\mathbf{H}_t\mathbf{x}_t(t_i) + \left(\mathbf{\Phi}_{k*} - \mathbf{B}_{dk*}\mathbf{G}^*_{ck*}\right)(\mathbf{I} - \mathbf{K}_{k*}\mathbf{H}_{k*})\hat{\mathbf{x}}_{k*}(t_i^-) \\ &+ \left(\mathbf{\Phi}_{k*} - \mathbf{B}_{dk*}\mathbf{G}^*_{ck*}\right)\mathbf{K}_{k*}\mathbf{v}_t(t_i)\end{aligned} \tag{4.40}$$

Now the state equations can be written conveniently in the augmented form as:

$$\begin{bmatrix}\mathbf{x}_t(t_{i+1}) \\ \hat{\mathbf{x}}_k(t_{i+1}^-) \\ \hat{\mathbf{x}}_{k*}(t_{i+1}^-)\end{bmatrix} = \begin{bmatrix}\left(\mathbf{\Phi}_t - \mathbf{B}_{dt}\mathbf{G}^*_{ck*}\mathbf{K}_{k*}\mathbf{H}_t\right) & \mathbf{0} & -\mathbf{B}_{dt}\mathbf{G}^*_{ck*}\left(\mathbf{I} - \mathbf{K}_{k*}\mathbf{H}_{k*}\right) \\ \left(\mathbf{\Phi}_k\mathbf{K}_k - \mathbf{B}_{dk}\mathbf{G}^*_{ck*}\mathbf{K}_{k*}\right)\mathbf{H}_t & \mathbf{\Phi}_k\left(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k\right) & -\mathbf{B}_{dk}\mathbf{G}^*_{ck*}\left(\mathbf{I} - \mathbf{K}_{k*}\mathbf{H}_{k*}\right) \\ \left(\mathbf{\Phi}_{k*} - \mathbf{B}_{dk*}\mathbf{G}^*_{ck*}\right)\mathbf{K}_{k*}\mathbf{H}_t & \mathbf{0} & \left(\mathbf{\Phi}_{k*} - \mathbf{B}_{dk*}\mathbf{G}^*_{ck*}\right)\left(\mathbf{I} - \mathbf{K}_{k*}\mathbf{H}_{k*}\right)\end{bmatrix}\begin{bmatrix}\mathbf{x}_t(t_i^-) \\ \hat{\mathbf{x}}_k(t_i) \\ \hat{\mathbf{x}}_{k*}(t_i^-)\end{bmatrix} + \begin{bmatrix}\mathbf{G}_{dt} & -\mathbf{B}_{dt}\mathbf{G}^*_{ck*}\mathbf{K}_{k*} \\ \mathbf{0} & \left(\mathbf{\Phi}_k\mathbf{K}_k - \mathbf{B}_{dk}\mathbf{G}^*_{ck*}\mathbf{K}_{k*}\right) \\ \mathbf{0} & \left(\mathbf{\Phi}_{k*} - \mathbf{B}_{dk*}\mathbf{G}^*_{ck*}\right)\mathbf{K}_{k*}\end{bmatrix}\begin{bmatrix}\mathbf{w}_{dt}(t_i) \\ \mathbf{v}_t(t_i)\end{bmatrix} \tag{4.41}$$

From Equation (4.41) define:

$$\mathbf{T}_{Gen} = \begin{bmatrix}\left(\mathbf{\Phi}_t - \mathbf{B}_{dt}\mathbf{G}^*_{ck*}\mathbf{K}_{k*}\mathbf{H}_t\right) & \mathbf{0} & -\mathbf{B}_{dt}\mathbf{G}^*_{ck*}\left(\mathbf{I} - \mathbf{K}_{k*}\mathbf{H}_{k*}\right) \\ \left(\mathbf{\Phi}_k\mathbf{K}_k - \mathbf{B}_{dk}\mathbf{G}^*_{ck*}\mathbf{K}_{k*}\right)\mathbf{H}_t & \mathbf{\Phi}_k\left(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k\right) & -\mathbf{B}_{dk}\mathbf{G}^*_{ck*}\left(\mathbf{I} - \mathbf{K}_{k*}\mathbf{H}_{k*}\right) \\ \left(\mathbf{\Phi}_{k*} - \mathbf{B}_{dk*}\mathbf{G}^*_{ck*}\right)\mathbf{K}_{k*}\mathbf{H}_t & \mathbf{0} & \left(\mathbf{\Phi}_{k*} - \mathbf{B}_{dk*}\mathbf{G}^*_{ck*}\right)\left(\mathbf{I} - \mathbf{K}_{k*}\mathbf{H}_{k*}\right)\end{bmatrix} \tag{4.42}$$

and

$$\mathbf{L}_{\text{Gen}} = \begin{bmatrix} \mathbf{G}_{\text{dt}} & -\mathbf{B}_{\text{dt}}\mathbf{G}^{*}_{\text{ck*}}\mathbf{K}_{\text{k*}} \\ \mathbf{0} & \left(\mathbf{\Phi}_{\text{k}}\mathbf{K}_{\text{k}} - \mathbf{B}_{\text{dk}}\mathbf{G}^{*}_{\text{ck*}}\mathbf{K}_{\text{k*}}\right) \\ \mathbf{0} & \left(\mathbf{\Phi}_{\text{k*}} - \mathbf{B}_{\text{dk*}}\mathbf{G}^{*}_{\text{ck*}}\right)\mathbf{K}_{\text{k*}} \end{bmatrix} \tag{4.43}$$

The state autocorrelation can now be written as:

$$\begin{aligned} &\mathrm{E}\left\{\begin{bmatrix} \mathbf{x}_{\text{t}}(t_{i+1}) \\ \hat{\mathbf{x}}_{\text{k}}(t_{i+1}^{-}) \\ \hat{\mathbf{x}}_{\text{k*}}(t_{i+1}^{-}) \end{bmatrix}\left[\mathbf{x}_{\text{t}}^{\text{T}}(t_{i+1}) \quad \hat{\mathbf{x}}_{\text{k}}^{\text{T}}(t_{i+1}^{-}) \quad \hat{\mathbf{x}}_{\text{k*}}^{\text{T}}(t_{i+1}^{-})\right]\right\} \\ &\qquad = \mathbf{\Xi}_{\text{k}}(t_{i+1}^{-}) = \mathbf{T}_{\text{Gen}}\,\mathbf{\Xi}_{\text{k}}(t_{i}^{-})\,\mathbf{T}_{\text{Gen}}^{\text{T}} + \mathbf{L}_{\text{Gen}}\mathbf{Q}_{\mathbf{0}}\mathbf{L}_{\text{Gen}}^{\text{T}} \end{aligned} \tag{4.44}$$

The output autocorrelation is given by $\mathrm{E}\{\mathbf{y}^{\text{T}}\mathbf{W}\,\mathbf{y}\} \equiv \mathrm{tr}(\mathbf{W}\,\mathbf{C}\,\mathbf{\Psi}_{\mathbf{x}_{\text{t}}}\mathbf{C}^{\text{T}})$, where $\mathbf{\Psi}_{\mathbf{x}_{\text{t}}} = \mathrm{E}\{[\mathbf{x}_{\text{t}}][\mathbf{x}_{\text{t}}]^{\text{T}}\}$ is the upper left partition of $\mathbf{\Xi}_{\text{k}\,\infty}$. Finally, the cost to be minimized is given by Equation (4.1).

4.4.2 Baram Distance Measure for the Generalized MMAC

In order to evaluate the performance measure in Equation (4.1), the selected controller has to be chosen, which of course is determined by the hypothesis conditional probability computation based on the residuals from the PFB portion of the GMMAC. As is done for the typical MMAC [56,57], the filter that is assumed to have the maximum probability at steady state is the filter with the minimum of the proximity measure given by Equation (4.12) and Equation (4.13). For the generalized MMAC, a modification is necessary in order to disregard the portion of the autocorrelation associated with the filters in the controller. This change yields:

$$\mathbf{N}_{\text{k}} = [\mathbf{H}_{\text{t}} \quad -\mathbf{H}_{\text{k}} \quad \mathbf{0}]\,\mathbf{\Xi}_{\text{k}\infty}[\mathbf{H}_{\text{t}} \quad -\mathbf{H}_{\text{k}} \quad \mathbf{0}]^{\text{T}} \tag{4.45}$$

where $\mathbf{\Xi}_{\text{k}\,\infty}$ is the state prediction autocorrelation computed by Equation (4.44) and $\mathbf{H}_{\text{k}}$ and $\mathbf{H}_{\text{t}}$ are the output matrix of the kth filter of the PFB and truth model, respectively.

4.4.3 Generalized MMAC Discretization

Discretization is defined as the determination of the optimal placement of filters in parameter space. For the conventional MMAC, the controller gain is designed based on the parameter locations. For the GMMAC, not only the controller gain, but also the state estimator for the control element is designed based on a different point in the parameter space from that assumed by the filter in the PFB. However, the architecture for the GMMAC is defined such that the control element is associated with a corresponding filter in the GMMAC. Hence, when one filter has assumed the maximum available probability, there is a corresponding control element that is active.

Since there is a one-to-one correspondence between the filter in the PFB and the control, the discretization still involves placing the filter of the PFB in parameter space, similar to the process used for the conventional MMAC. At the parameter location for the filter, a generalized LQG controller is designed. Thus, the discretization algorithm follows similar steps as the conventional MMAC from Chapter 2 with the modification from Section 4.2, and is summarized as follows:

1) Describe in terms of the parameter vector $\mathbf{a}$, the truth model of the system, the filter in the PFB and the controller element.
2) Choose the number of filters K in the PFB.
3) Choose a representative parameter set, $\{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_K\}$ to begin the minimization.
4) For each parameter in the representative parameter set $\{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_K\}$, design a generalized LQG elemental controller.

5) Use a numerical integration technique to evaluate the cost function given by Equation (4.1).

   a) Compute $\boldsymbol{\Xi}_{k\,\infty}$ using Equation (4.44) at discrete points in the parameter space (for $\mathbf{a}_t$ value) for each filter: k = 1,…K and the corresponding generalized LQG controller element.

   b) At each discrete point, evaluate the proximity measure using Equation (4.12) and Equation (4.13) with the modification from Equation (4.45), and $\boldsymbol{\Xi}_{k\,\infty}$ for k = 1,…K, to determine the convergence to a single filter.

   c) For the selected elemental filter, determine $\boldsymbol{\Psi}_{\mathbf{x}_t}$ from the previous evaluation of $\boldsymbol{\Xi}_{k\,\infty}$ for k = 1,…K and use in the evaluation of the cost function.

6) Use a vector minimization technique with the functional evaluation from Step (5) to minimize $J_{2\,c}$.

This discretization algorithm can be slightly modified in order to use the PFB to provide the best parameter estimate. Rather than trying to place the filters in the PFB for best control, the filters in step two are set according to the discretization for optimal parameter estimation as reviewed in Chapter 2. The remaining steps for determining the optimal LQG controller remain the same.

4.4.4 Lower Bounding of the Probability for the Generalized MMAC

As is the case with the typical MMAC, it is necessary to place a lower bound on the probability weight that is assigned to each controller element in order to prevent probability lock-out. In order to account for the lower bound on the probability assigned to the controller, the performance equations need to include the probability assigned to each controller element. To implement this lower bound, assume all the controllers will have a probability weight of p$_{\min}$ except for one controller. That controller will have a probability expressed as:

$$p_{\text{sel}} = 1 - p_{\min}(\text{K-1}) \tag{4.46}$$

For the GMMAC, the probability assigned to each controller element is derived from the filters in the PFB. To yield the control, the probability is then assigned to each controller element as:

$$\mathbf{u}_{\text{k}*}(t_i) = -p_1\mathbf{G}^*_{\text{c}1*}\hat{\mathbf{x}}_{1*}(t_i^+) - p_2\mathbf{G}^*_{\text{c}2*}\hat{\mathbf{x}}_{2*}(t_i^+) \ \ldots \ - p_{\text{K}}\mathbf{G}^*_{\text{cK}*}\hat{\mathbf{x}}_{\text{K}*}(t_i^+) \tag{4.47}$$

Since this control is input to the filters in the PFB, the state equations will be expressed as:

$$\hat{\mathbf{x}}_{\text{k}}(t_{i+1}^-) = \mathbf{\Phi}_{\text{k}}\hat{\mathbf{x}}_{\text{k}}(t_i^+) - \mathbf{B}_{\text{d k}}\left(p_1\mathbf{G}^*_{\text{c}1*}\hat{\mathbf{x}}_{1*}(t_i^+) + p_2\mathbf{G}^*_{\text{c}2*}\hat{\mathbf{x}}_{2*}(t_i^+) \ \ldots \ + p_{\text{K}}\mathbf{G}^*_{\text{cK}*}\hat{\mathbf{x}}_{\text{K}*}(t_i^+)\right) \tag{4.48}$$

Now substitute the filter update equation and expand to derive:

$$\begin{aligned}\hat{\mathbf{x}}_{\text{k}}(t_{i+1}^-) = \mathbf{\Phi}_{\text{k}}&\left((\mathbf{I} - \mathbf{K}_{\text{k}}\mathbf{H}_{\text{k}})\hat{\mathbf{x}}_{\text{k}}(t_{\text{i}}^-) + \mathbf{K}_{\text{k}}\mathbf{H}_{\text{t}}\mathbf{x}_{\text{t}}(t_{\text{i}}) + \mathbf{K}_{\text{k}}\mathbf{v}_{\text{t}}(t_{\text{i}})\right)\\ &- p_1\mathbf{B}_{\text{d k}}\mathbf{G}^*_{\text{c}1*}\left((\mathbf{I} - \mathbf{K}_{1*}\mathbf{H}_{1*})\hat{\mathbf{x}}_{1*}(t_{\text{i}}^-) + \mathbf{K}_{1*}\mathbf{H}_{\text{t}}\mathbf{x}_{\text{t}}(t_{\text{i}}) + \mathbf{K}_{1*}\mathbf{v}_{\text{t}}(t_{\text{i}})\right)\\ &- p_2\mathbf{B}_{\text{d k}}\mathbf{G}^*_{\text{c}2*}\left((\mathbf{I} - \mathbf{K}_{2*}\mathbf{H}_{2*})\hat{\mathbf{x}}_{2*}(t_{\text{i}}^-) + \mathbf{K}_{2*}\mathbf{H}_{\text{t}}\mathbf{x}_{\text{t}}(t_{\text{i}}) + \mathbf{K}_{2*}\mathbf{v}_{\text{t}}(t_{\text{i}})\right) \ldots\\ &- p_{\text{K}}\mathbf{B}_{\text{d k}}\mathbf{G}^*_{\text{cK}*}\left((\mathbf{I} - \mathbf{K}_{\text{K}*}\mathbf{H}_{\text{K}*})\hat{\mathbf{x}}_{\text{K}*}(t_{\text{i}}^-) + \mathbf{K}_{\text{K}*}\mathbf{H}_{\text{t}}\mathbf{x}_{\text{t}}(t_{\text{i}}) + \mathbf{K}_{\text{K}*}\mathbf{v}_{\text{t}}(t_{\text{i}})\right)\end{aligned} \tag{4.49}$$

To develop the equations for the Kalman filter in the controller element, the control from Equation (4.48) is substituted into the propagation equations to yield:

$$\hat{\mathbf{x}}_{\text{k}*}(t_{i+1}^-) = \mathbf{\Phi}_{\text{k}*}\hat{\mathbf{x}}_{\text{k}*}(t_i^+) - \mathbf{B}_{\text{d k}*}\left(p_1\mathbf{G}^*_{\text{c}1*}\hat{\mathbf{x}}_{1*}(t_i^+) + p_2\mathbf{G}^*_{\text{c}2*}\hat{\mathbf{x}}_{2*}(t_i^+) \ \ldots \ + p_{\text{K}}\mathbf{G}^*_{\text{cK}*}\hat{\mathbf{x}}_{\text{K}*}(t_i^+)\right) \tag{4.50}$$

Now incorporate the filter update equation and expand to yield:

$$\begin{aligned}\hat{\mathbf{x}}_{\text{k}*}(t_{i+1}^-) = \mathbf{\Phi}_{\text{k}*}&\left((\mathbf{I} - \mathbf{K}_{\text{k}*}\mathbf{H}_{\text{k}*})\hat{\mathbf{x}}_{\text{k}*}(t_{\text{i}}^-) + \mathbf{K}_{\text{k}*}\mathbf{H}_{\text{t}}\mathbf{x}_{\text{t}}(t_{\text{i}}) + \mathbf{K}_{\text{k}*}\mathbf{v}_{\text{t}}(t_{\text{i}})\right)\\ &- p_1\mathbf{B}_{\text{d k}*}\mathbf{G}^*_{\text{c}1*}\left((\mathbf{I} - \mathbf{K}_{1*}\mathbf{H}_{1*})\hat{\mathbf{x}}_{1*}(t_{\text{i}}^-) + \mathbf{K}_{1*}\mathbf{H}_{\text{t}}\mathbf{x}_{\text{t}}(t_{\text{i}}) + \mathbf{K}_{1*}\mathbf{v}_{\text{t}}(t_{\text{i}})\right)\\ &- p_2\mathbf{B}_{\text{d k}*}\mathbf{G}^*_{\text{c}2*}\left((\mathbf{I} - \mathbf{K}_{2*}\mathbf{H}_{2*})\hat{\mathbf{x}}_{2*}(t_{\text{i}}^-) + \mathbf{K}_{2*}\mathbf{H}_{\text{t}}\mathbf{x}_{\text{t}}(t_{\text{i}}) + \mathbf{K}_{2*}\mathbf{v}_{\text{t}}(t_{\text{i}})\right) \ldots\\ &- p_K\mathbf{B}_{\text{d k}*}\mathbf{G}^*_{\text{cK}*}\left((\mathbf{I} - \mathbf{K}_{\text{K}*}\mathbf{H}_{\text{K}*})\hat{\mathbf{x}}_{\text{K}*}(t_{\text{i}}^-) + \mathbf{K}_{\text{K}*}\mathbf{H}_{\text{t}}\mathbf{x}_{\text{t}}(t_{\text{i}}) + \mathbf{K}_{\text{K}*}\mathbf{v}_{\text{t}}(t_{\text{i}})\right)\end{aligned} \tag{4.51}$$

Finally, the propagation equation for the true system with the control from Equation (4.48) substituted into the expression is given by:

$$\begin{aligned}\mathbf{x}_{\mathrm{t}}(t_{i+1}) = \mathbf{\Phi}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) - \mathbf{B}_{\mathrm{d\,t}}\left(p_1\mathbf{G}^*_{\mathrm{c}1*}\hat{\mathbf{x}}_{1*}(t_i^+) + p_2\mathbf{G}^*_{\mathrm{c}2*}\hat{\mathbf{x}}_{2*}(t_i^+)\ \dots\ + p_{\mathrm{K}}\mathbf{G}^*_{\mathrm{cK}*}\hat{\mathbf{x}}_{\mathrm{K}*}(t_i^+)\right) \\ + \mathbf{G}_{\mathrm{d\,t}}\mathbf{w}_{\mathrm{d\,t}}(t_i)\end{aligned} \tag{4.52}$$

Now substitute the filter update equation into that result and expand to yield:

$$\begin{aligned}\mathbf{x}_{\mathrm{t}}(t_{i+1}) &= \mathbf{\Phi}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{\mathrm{i}}) \\ &- p_1\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c}1*}\left((\mathbf{I}-\mathbf{K}_{1*}\mathbf{H}_{1*})\hat{\mathbf{x}}_{1*}(t_{\mathrm{i}}^-) + \mathbf{K}_{1*}\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{\mathrm{i}}) + \mathbf{K}_{1*}\mathbf{v}_{\mathrm{t}}(t_{\mathrm{i}})\right) \\ &- p_2\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c}2*}\left((\mathbf{I}-\mathbf{K}_{2*}\mathbf{H}_{2*})\hat{\mathbf{x}}_{2*}(t_{\mathrm{i}}^-) + \mathbf{K}_{2*}\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{\mathrm{i}}) + \mathbf{K}_{2*}\mathbf{v}_{\mathrm{t}}(t_{\mathrm{i}})\right) \dots \\ &- p_{\mathrm{K}}\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{cK}*}\left((\mathbf{I}-\mathbf{K}_{\mathrm{K}*}\mathbf{H}_{\mathrm{K}*})\hat{\mathbf{x}}_{\mathrm{K}*}(t_{\mathrm{i}}^-) + \mathbf{K}_{\mathrm{K}*}\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_{\mathrm{i}}) + \mathbf{K}_{\mathrm{K}*}\mathbf{v}_{\mathrm{t}}(t_{\mathrm{i}})\right) \\ &+ \mathbf{G}_{\mathrm{d\,t}}\mathbf{w}_{\mathrm{d\,t}}(t_i)\end{aligned} \tag{4.53}$$

Simplification yields:

$$\begin{aligned}\mathbf{x}_{\mathrm{t}}(t_{i+1}) = (\mathbf{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{d\,t}}\sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}}\mathbf{G}^*_{\mathrm{c\,j}*}\mathbf{K}_{\mathrm{j}*}\mathbf{H}_{\mathrm{t}})\mathbf{x}_{\mathrm{t}}(t_i) - p_1\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c}1}(\mathbf{I}-\mathbf{K}_{1*}\mathbf{H}_1)\hat{\mathbf{x}}_{1*}(t_i^-)\ \dots \\ - p_{\mathrm{K}}\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{cK}*}(\mathbf{I}-\mathbf{K}_{\mathrm{K}*}\mathbf{H}_{\mathrm{K}*})\hat{\mathbf{x}}_{\mathrm{K}*}(t_i^-) - (\mathbf{B}_{\mathrm{d\,t}}\sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}}\mathbf{G}^*_{\mathrm{c\,j}*}\mathbf{K}_{\mathrm{j}*})\mathbf{v}_{\mathrm{t}}(t_{\mathrm{i}}) \\ + \mathbf{G}_{\mathrm{d\,t}}\mathbf{w}_{\mathrm{d\,t}}(t_i)\end{aligned} \tag{4.54}$$

Now, the augmented state equations that form the one-step prediction model for the GMMAC use Equations (4.49), (4.51), and (4.54) and is expressed as:

$$
\begin{bmatrix}
\mathbf{x}_{\mathrm{t}}(t_{i+1}) \\
\hat{\mathbf{x}}_1(t_{i+1}^-) \\
\vdots \\
\hat{\mathbf{x}}_{\mathrm{K}}(t_{i+1}^-) \\
\hat{\mathbf{x}}_{1*}(t_{i+1}^-) \\
\vdots \\
\hat{\mathbf{x}}_{\mathrm{K}*}(t_{i+1}^-)
\end{bmatrix}
=
\left[
\begin{matrix}
\mathbf{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{d\,t}} \sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}_{\mathrm{c\,j}}^{*} \mathbf{K}_{\mathrm{j}} \mathbf{H}_{\mathrm{t}} \\
\mathbf{\Phi}_1 \mathbf{K}_1 \mathbf{H}_{\mathrm{t}} - \mathbf{B}_{\mathrm{d}1} \sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}_{\mathrm{c\,j*}}^{*} \mathbf{K}_{\mathrm{j}} \mathbf{H}_{\mathrm{t}} \\
\vdots \\
\mathbf{\Phi}_{\mathrm{K}} \mathbf{K}_{\mathrm{K}} \mathbf{H}_{\mathrm{t}} - \mathbf{B}_{\mathrm{d\,K}} \sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}_{\mathrm{c\,j*}}^{*} \mathbf{K}_{\mathrm{j}} \mathbf{H}_{\mathrm{t}} \\
\mathbf{\Phi}_{1*} \mathbf{K}_{1*} \mathbf{H}_{\mathrm{t}} - \mathbf{B}_{\mathrm{d}1*} \sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}_{\mathrm{c\,j*}}^{*} \mathbf{K}_{\mathrm{j*}} \mathbf{H}_{\mathrm{t}} \\
\vdots \\
\mathbf{\Phi}_{\mathrm{K}*} \mathbf{K}_{\mathrm{K}*} \mathbf{H}_{\mathrm{t}} - \mathbf{B}_{\mathrm{d\,K}*} \sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}_{\mathrm{c\,j*}}^{*} \mathbf{K}_{\mathrm{j*}} \mathbf{H}_{\mathrm{t}}
\end{matrix}
\right.
$$

$$
\begin{matrix}
\mathbf{0} & \cdots & \mathbf{0} \\
\mathbf{\Phi}_1(\mathbf{I} - \mathbf{K}_1 \mathbf{H}_1) & \cdots & \mathbf{0} \\
\vdots & \ddots & \vdots \\
\mathbf{0} & \cdots & \mathbf{\Phi}_{\mathrm{K}}(\mathbf{I} - \mathbf{K}_{\mathrm{K}} \mathbf{H}_{\mathrm{K}}) \\
\mathbf{0} & \cdots & \mathbf{0} \\
\vdots & \vdots & \vdots \\
\mathbf{0} & \cdots & \mathbf{0}
\end{matrix}
$$

$$
\left.
\begin{matrix}
-p_1 \mathbf{B}_{\mathrm{d\,t}} \mathbf{G}_{\mathrm{c}1*}^{*} (\mathbf{I} - \mathbf{K}_{1*} \mathbf{H}_1) & \cdots & -p_{\mathrm{K}} \mathbf{B}_{\mathrm{d\,t}} \mathbf{G}_{\mathrm{c\,K}*}^{*} (\mathbf{I} - \mathbf{K}_{\mathrm{K}*} \mathbf{H}_{\mathrm{K}*}) \\
-p_1 \mathbf{B}_{\mathrm{d}1} \mathbf{G}_{\mathrm{c}1*}^{*} (\mathbf{I} - \mathbf{K}_{1*} \mathbf{H}_{1*}) & \cdots & -p_{\mathrm{K}} \mathbf{B}_{\mathrm{d\,K}} \mathbf{G}_{\mathrm{c\,K}*}^{*} (\mathbf{I} - \mathbf{K}_{\mathrm{K}*} \mathbf{H}_{\mathrm{K}*}) \\
\vdots & \ddots & \vdots \\
-p_1 \mathbf{B}_{\mathrm{d\,K}} \mathbf{G}_{\mathrm{c}\,1*}^{*} (\mathbf{I} - \mathbf{K}_{1*} \mathbf{H}_{1*}) & \cdots & -p_{\mathrm{K}} \mathbf{B}_{\mathrm{d\,K}} \mathbf{G}_{\mathrm{c\,K}*}^{*} (\mathbf{I} - \mathbf{K}_{\mathrm{K}*} \mathbf{H}_{\mathrm{K}*}) \\
(\mathbf{\Phi}_{1*} - p_1 \mathbf{B}_{\mathrm{d}1} \mathbf{G}_{\mathrm{c}1*}^{*})(\mathbf{I} - \mathbf{K}_{1*} \mathbf{H}_{1*}) & \cdots & -p_{\mathrm{K}} \mathbf{B}_{\mathrm{d\,K}} \mathbf{G}_{\mathrm{c\,K}*}^{*} (\mathbf{I} - \mathbf{K}_{\mathrm{K}*} \mathbf{H}_{\mathrm{K}*}) \\
\vdots & \ddots & \vdots \\
-p_1 \mathbf{B}_{\mathrm{d}1} \mathbf{G}_{\mathrm{c}1*}^{*} (\mathbf{I} - \mathbf{K}_{1*} \mathbf{H}_{1*}) & \cdots & (\mathbf{\Phi}_{\mathrm{K}*} - p_{\mathrm{K}} \mathbf{B}_{\mathrm{d\,K}} \mathbf{G}_{\mathrm{c\,K}*}^{*})(\mathbf{I} - \mathbf{K}_{\mathrm{K}*} \mathbf{H}_{\mathrm{K}*})
\end{matrix}
\right]
\begin{bmatrix}
\mathbf{x}_{\mathrm{t}}(t_i) \\
\hat{\mathbf{x}}_1(t_i^-) \\
\vdots \\
\hat{\mathbf{x}}_{\mathrm{K}}(t_i^-) \\
\hat{\mathbf{x}}_{1*}(t_i^-) \\
\vdots \\
\hat{\mathbf{x}}_{\mathrm{K}*}(t_i^-)
\end{bmatrix}
$$

$$
+ \begin{bmatrix}
-\mathbf{B}_{\mathrm{d\,t}} \mathbf{G}_{\mathrm{c\,nom}}^{*} \sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{K}^{\mathrm{j}} \\
(\mathbf{\Phi}_1 \mathbf{K}_1 - \mathbf{B}_{\mathrm{d}1} \sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}_{\mathrm{c\,j*}}^{*} \mathbf{K}_{\mathrm{j}}) \\
\vdots \\
(\mathbf{\Phi}_{\mathrm{K}} \mathbf{K}_{\mathrm{K}} - \mathbf{B}_{\mathrm{d\,K}} \sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}_{\mathrm{c\,j*}}^{*} \mathbf{K}_{\mathrm{j}}) \\
(\mathbf{\Phi}_{1*} \mathbf{K}_{1*} - \mathbf{B}_{\mathrm{d}1*} \sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}_{\mathrm{c\,j*}}^{*} \mathbf{K}_{\mathrm{j*}}) \\
\vdots \\
(\mathbf{\Phi}_{\mathrm{K}*} \mathbf{K}_{\mathrm{K}*} - \mathbf{B}_{\mathrm{d\,K}*} \sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}_{\mathrm{c\,j*}}^{*} \mathbf{K}_{\mathrm{j*}})
\end{bmatrix}
\mathbf{v}_{\mathrm{t}}(t_i) +
\begin{bmatrix}
\mathbf{G}_{\mathrm{d\,t}} \\
\mathbf{0} \\
\vdots \\
\mathbf{0} \\
\mathbf{0} \\
\vdots \\
\mathbf{0}
\end{bmatrix}
\mathbf{w}_{\mathrm{d\,t}}(t_i)
\qquad (4.55)
$$

Now from Equation (4.55) define:

$$\mathbf{T}_{\mathrm{GenLB}} = \left[\begin{array}{ccccccc}
\mathbf{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{dt}} \sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}^{*}_{\mathrm{cj}} \mathbf{K}_{\mathrm{j}} \mathbf{H}_{\mathrm{t}} & \mathbf{0} & \cdots & \mathbf{0} & -p_1 \mathbf{B}_{\mathrm{dt}} \mathbf{G}^{*}_{\mathrm{c1*}} (\mathbf{I} - \mathbf{K}_{1*} \mathbf{H}_1) & \cdots & -p_{\mathrm{K}} \mathbf{B}_{\mathrm{dt}} \mathbf{G}^{*}_{\mathrm{cK*}} (\mathbf{I} - \mathbf{K}_{\mathrm{K*}} \mathbf{H}_{\mathrm{K*}}) \\
\mathbf{\Phi}_1 \mathbf{K}_1 \mathbf{H}_{\mathrm{t}} - \mathbf{B}_{\mathrm{d1}} \sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}^{*}_{\mathrm{cj*}} \mathbf{K}_{\mathrm{j}} \mathbf{H}_{\mathrm{t}} & \mathbf{\Phi}_1 (\mathbf{I} - \mathbf{K}_1 \mathbf{H}_1) & \cdots & \mathbf{0} & -p_1 \mathbf{B}_{\mathrm{d1}} \mathbf{G}^{*}_{\mathrm{c1*}} (\mathbf{I} - \mathbf{K}_{1*} \mathbf{H}_{1*}) & \cdots & -p_{\mathrm{K}} \mathbf{B}_{\mathrm{dK}} \mathbf{G}^{*}_{\mathrm{cK*}} (\mathbf{I} - \mathbf{K}_{\mathrm{K*}} \mathbf{H}_{\mathrm{K*}}) \\
\vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
\mathbf{\Phi}_{\mathrm{K}} \mathbf{K}_{\mathrm{K}} \mathbf{H}_{\mathrm{t}} - \mathbf{B}_{\mathrm{dK}} \sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}^{*}_{\mathrm{cj*}} \mathbf{K}_{\mathrm{j}} \mathbf{H}_{\mathrm{t}} & \mathbf{0} & \cdots & \mathbf{\Phi}_{\mathrm{K}} (\mathbf{I} - \mathbf{K}_{\mathrm{K}} \mathbf{H}_{\mathrm{K}}) & -p_1 \mathbf{B}_{\mathrm{dK}} \mathbf{G}^{*}_{\mathrm{c1*}} (\mathbf{I} - \mathbf{K}_{1*} \mathbf{H}_{1*}) & \cdots & -p_{\mathrm{K}} \mathbf{B}_{\mathrm{dK}} \mathbf{G}^{*}_{\mathrm{cK*}} (\mathbf{I} - \mathbf{K}_{\mathrm{K*}} \mathbf{H}_{\mathrm{K*}}) \\
\mathbf{\Phi}_{1*} \mathbf{K}_{1*} \mathbf{H}_{\mathrm{t}} - \mathbf{B}_{\mathrm{d1*}} \sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}^{*}_{\mathrm{cj*}} \mathbf{K}_{\mathrm{j*}} \mathbf{H}_{\mathrm{t}} & \mathbf{0} & \cdots & \mathbf{0} & (\mathbf{\Phi}_{1*} - p_1 \mathbf{B}_{\mathrm{d1}} \mathbf{G}^{*}_{\mathrm{c1*}})(\mathbf{I} - \mathbf{K}_{1*} \mathbf{H}_{1*}) & \cdots & -p_{\mathrm{K}} \mathbf{B}_{\mathrm{dK}} \mathbf{G}^{*}_{\mathrm{cK*}} (\mathbf{I} - \mathbf{K}_{\mathrm{K*}} \mathbf{H}_{\mathrm{K*}}) \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
\mathbf{\Phi}_{\mathrm{K*}} \mathbf{K}_{\mathrm{K*}} \mathbf{H}_{\mathrm{t}} - \mathbf{B}_{\mathrm{dK*}} \sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}} \mathbf{G}^{*}_{\mathrm{cj*}} \mathbf{K}_{\mathrm{j*}} \mathbf{H}_{\mathrm{t}} & \mathbf{0} & \cdots & \mathbf{0} & -p_1 \mathbf{B}_{\mathrm{d1}} \mathbf{G}^{*}_{\mathrm{c1*}} (\mathbf{I} - \mathbf{K}_{1*} \mathbf{H}_{1*}) & \cdots & (\mathbf{\Phi}_{\mathrm{K*}} - p_{\mathrm{K}} \mathbf{B}_{\mathrm{dK}} \mathbf{G}^{*}_{\mathrm{cK*}})(\mathbf{I} - \mathbf{K}_{\mathrm{K*}} \mathbf{H}_{\mathrm{K*}})
\end{array}\right] \tag{4.56}$$

and also define:

$$\mathbf{L}_{\mathrm{GenLB}} = \begin{bmatrix} \mathbf{G}_{\mathrm{d\,t}} & -\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^{*}_{\mathrm{c\,nom}}\sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}}\mathbf{K}^{\mathrm{j}} \\ \mathbf{0} & (\mathbf{\Phi}_{1}\mathbf{K}_{1} - \mathbf{B}_{\mathrm{d1}}\sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}}\mathbf{G}^{*}_{\mathrm{c\,j*}}\mathbf{K}_{\mathrm{j}}) \\ \vdots & \vdots \\ \mathbf{0} & (\mathbf{\Phi}_{\mathrm{K}}\mathbf{K}_{\mathrm{K}} - \mathbf{B}_{\mathrm{d\,K}}\sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}}\mathbf{G}^{*}_{\mathrm{c\,j*}}\mathbf{K}_{\mathrm{j}}) \\ \mathbf{0} & (\mathbf{\Phi}_{1*}\mathbf{K}_{1*} - \mathbf{B}_{\mathrm{d1*}}\sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}}\mathbf{G}^{*}_{\mathrm{c\,j*}}\mathbf{K}_{\mathrm{j*}}) \\ \vdots & \vdots \\ \mathbf{0} & (\mathbf{\Phi}_{\mathrm{K*}}\mathbf{K}_{\mathrm{K*}} - \mathbf{B}_{\mathrm{d\,K*}}\sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}}\mathbf{G}^{*}_{\mathrm{c\,j*}}\mathbf{K}_{\mathrm{j*}}) \end{bmatrix} \tag{4.57}$$

By substituting $\mathbf{T}_{\mathrm{GenLB}}$ and $\mathbf{L}_{\mathrm{GenLB}}$ for $\mathbf{T}_{\mathrm{Gen}}$ and $\mathbf{L}_{\mathrm{Gen}}$ respectively, Equation (4.44) computes the state autocorrelation, $\mathbf{\Xi}_{\mathrm{k}\,\infty}$. The output autocorrelation used in the cost evaluation is given by $\mathrm{E}\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\,\mathbf{y}\} \equiv \mathrm{tr}(\mathbf{W}\,\mathbf{C}\,\mathbf{\Psi}_{\mathbf{x}_{\mathrm{t}}}\mathbf{C}^{\mathrm{T}})$, and $\mathbf{\Psi}_{\mathbf{x}_{\mathrm{t}}} = \mathrm{E}\{[\mathbf{x}_{\mathrm{t}}][\mathbf{x}_{\mathrm{t}}]^{\mathrm{T}}\}$ is given by the upper left partition of $\mathbf{\Xi}_{\mathrm{k}\,\infty}$ as before. Additionally, the discretization process for the parameter space will not change from the previous section. It simply uses the cost evaluation as just described.

4.4.5 LQG Controller with Enhanced Robustness Control Element

A variation to the GMMAC is the addition of enhanced robustness to the control element. Recall that, for the modified MMAC with robustness from the previous section, the robustness region was centered about the point that specifies the filter for the state estimate. The gain control matrix had to be designed for the desired robustness around this point and using the filter from the MME. However, for the modified LQG control with robustness from Chapter 3, the assumed true system was considered the center point around which the designer specifies the radius of robustness. Consequently, the design method finds the best filter and controller to provide the best performance according to the performance measure for robustness. The assumed truth model, filter model, and controller model may all be different. Although this is not possible for the modified MMAC with enhanced robustness, it is possible to have different models with the GMMAC with a robust control element approach.

The derivation of the GMMAC with a robust LQG control element entails a combination of the previous MMAC enhancements. Similar to the MMAC with enhanced robustness approach from Section 4.3, Equation (4.23) that describes the cost function to be minimized is the same for the GMMAC with a robust LQG control element. This cost function captures the region of robustness associated with the control elements and the assumed placement of the filters in parameter space. The evaluation of the performance can be taken directly from the GMMAC derivation in Equations (4.41) through (4.44) and Equations (4.55) through (4.57) for implementation of lower bounding. The filter state equations for the PFB reflect the center point about which the designer specifies the radius of robustness. The LQG state equations reflect the LQG control design portion. Specifically, $\mathbf{K}_{j*}$ and $\mathbf{G}^{*}_{cj*}$ specify the jth Kalman filter gain for the state estimation and the full-state feedback controller gain, respectively.

Figure 4.5 illustrates the possible separation of the LQG Kalman filter model and

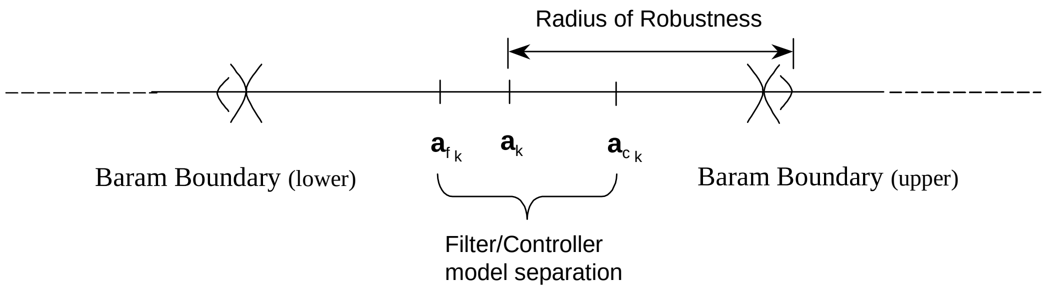

Figure 4.5 Robustness region covered by controller and filter with different models.

controller model from the assumed PFB filter model. The discretization algorithm will determine the PFB filters' parameter value locations and the radius of robustness. The radius of robustness will determine the Filter/Controller location of the enhanced LQG controller element. The Baram boundary does not necessarily limit the radius of robustness. As determined by the optimization, a more conservative coverage of the parameter space might extend the radius of robustness beyond the Baram boundary. On the other hand, complete coverage of the Baram boundaries may not provide the minimal cost and the radius of robustness might be less than the boundary distance. As usual, an average performance over a region will require some points to suffer in performance in return for improvements elsewhere.

The optimization process guarantees stability. The cost will be minimized and any design that yields an unstable performance will be disregarded. The design is for the closed-loop control in which the MMAE portion is integrated into the complete MMAC design. Such is the case in the conventional MMAC for which the optimization places the LQG controllers (without enhanced robustness) so that the cost is minimized. A successful optimization would not allow for designs that are unstable. If the MMAE portion of the GMMAC were designed without regard for the controllers, then to guarantee stability, the controllers would have to be designed such that the robustness region covers the whole Baram boundary set by the MMAE.

The discretization algorithm for the GMMAC with robust control element requires only one modification to the one used for the Modified MMAC with enhanced robustness equation given in Section 4.3.3. The change involves specifying a modified

LQG controller with enhanced robustness to be used in the cost evaluation given in Equation (4.23). This modification is:

4 a) Design the modified LQG controller with enhanced robustness for the parameter set **a** and the corresponding robustness of radius **r**.

As before, the discretization algorithm using Equation (4.23) does not require that the radius of robustness be specified prior to the discretization. The controller's robustness will affect the filter locations in the PFB and vice versa. The optimization will produce the optimal filter locations for the PFB and enhanced robustness of the controller element.

4.5 Variance of the Proximity Measure

Since models in the bank of filters may not match the true system, the proximity measure developed by Baram and discussed in Chapter 2 is used to determine the closest filter in probability for computing predicted performance. As discussed previously, the performance prediction algorithm assumes that the MME bank (or PFB) selects one filter and does not consider any blending that may occur before the system reaches steady state conditions. The amount of blending depends on "how fast" the filter bank reaches steady state such that one filter has assumed all the probability. Of course, blending also depends on lower bounds on probability placed on the filters in the bank and used to prevent probability lockout. The factors contribute to *how fast* the filter bank reaches steady state is yet to be determined. This section attempts to show that performance can be related to the variance of the proximity measure.

The Baram measure uses the covariance of the residuals. Since there is a *spread* in the residuals, at points in parameter space around the Baram boundary, the same filter will not be selected from one simulation to the next, given that there are different noise histories. This variation in selection of the filter will affect performance. Thus, it is desirable to predict this variation and include it in the performance prediction algorithms.

The spread of the distribution of the predicted filter models chosen by the proximity measure about the true system model is essentially the variance of the proximity measure. The variance of the proximity measure, which is derived from the covariance of the residuals, essentially equates to the fourth moment of the residuals.

4.5.1 Computation

To compute the variance of the proximity measure is to compute the variance of $\mathbf{r}_k^T \mathbf{A}_k^{-1} \mathbf{r}_k$. With substitution, the variance is expressed as:

$$P_{\mathbf{r}_k^T \mathbf{A}_k^{-1} \mathbf{r}_k} = E\left\{\left(\mathbf{r}_k^T \mathbf{A}_k^{-1} \mathbf{r}_k\right)^2\right\} - E\left\{\left(\mathbf{r}_k^T \mathbf{A}_k^{-1} \mathbf{r}_k\right)\right\}^2 \tag{4.58}$$

From Baram's work and as used in Sheldon's, the predicted mean value of $\mathbf{r}_k{}^T \mathbf{A}_k^{-1} \mathbf{r}_k$ (directly related to the second moment of the residual vector $\mathbf{r}^*$) in the kth filter is given by:

$$E\left\{\mathbf{r}_k^T \mathbf{A}_k^{-1} \mathbf{r}_k\right\} = \mathrm{trace}\left(\mathbf{M}_k^{-1} \mathbf{N}_k\right) \tag{4.59}$$

where

$$\mathbf{N}_k = \left[-\mathbf{H}_k \quad \mathbf{H}_t\right] \mathbf{\Xi}_{k\infty} \left[-\mathbf{H}_k \quad \mathbf{H}_t\right]^T \tag{4.60}$$

and

$$\mathbf{M}_k = \mathbf{H}_k \mathbf{P}_k^{-} \mathbf{H}_k{}^T + \mathbf{R}_k \tag{4.61}$$

For the case of scalar measurements, Equation (4.59) reduces to :

$$E\left\{r_k^T A_k^{-1} r_k\right\} = N_k \big/ M_k \tag{4.62}$$

To compute the autocorrelation of $\mathbf{r}_k^T \mathbf{A}_k^{-1} \mathbf{r}_k$, directly use the result from Maybeck [33] to obtain the following:

$$\mathrm{E}\left\{\left(\mathbf{r}_k{}^T \mathbf{A}_k^{-1} \mathbf{r}_k\right)^2\right\} = \left[\mathrm{trace}\left(\mathbf{M}_k^{-1} \mathbf{N}_k\right)\right]^2 + 2\,\mathrm{trace}\left(\mathbf{M}_k^{-1} \mathbf{N}_k \mathbf{M}_k^{-1} \mathbf{N}_k\right) \tag{4.63}$$

For the case of scalar measurements, this reduces to:

$$\mathrm{E}\left\{\left(\mathbf{r}_k{}^T \mathbf{A}_k^{-1} \mathbf{r}_k\right)^2\right\} = 3\left({}^{\mathrm{N}_k}\!/_{\mathrm{M}_k}\right)^2 \tag{4.64}$$

Now substitute Equations (4.59) and (4.63) into Equations (4.58) to obtain the final form of the variance:

$$\begin{aligned}\sigma^2_{\mathbf{r}_k{}^T \mathbf{A}_k^{-1} \mathbf{r}_k} &= \left[\mathrm{trace}\left(\mathbf{M}_k^{-1} \mathbf{N}_k\right)\right]^2 + 2\,\mathrm{trace}\left(\mathbf{M}_k^{-1} \mathbf{N}_k \mathbf{M}_k^{-1} \mathbf{N}_k\right) \\ &\quad - \left[\mathrm{trace}\left(\mathbf{M}_k^{-1} \mathbf{N}_k\right)\right]^2 \\ &= 2\,\mathrm{trace}\left(\mathbf{M}_k^{-1} \mathbf{N}_k \mathbf{M}_k^{-1} \mathbf{N}_k\right)\end{aligned} \tag{4.65}$$

For the scalar-measurement case, this reduces to the simple form

$$\sigma^2_{\mathbf{r}_k{}^T \mathbf{A}_k^{-1} \mathbf{r}_k} = 2\left({}^{\mathrm{N}_k}\!/_{\mathrm{M}_k}\right)^2 \tag{4.66}$$

Continuing with the scalar-measurement case, the computation of the predicted spread of the proximity measure for a one-sigma bound becomes:

$$\ell_k \equiv \left(\log\left|\mathrm{M}_k\right| + {}^{\mathrm{N}_k}\!/_{\mathrm{M}_k}\right) \pm \sqrt{2}\left({}^{\mathrm{N}_k}\!/_{\mathrm{M}_k}\right) \tag{4.67}$$

4.5.2 Analysis

The effects of the variation of filter selection on the proximity measure will be most notable when the true system approaches the Baram boundary. This boundary is defined as the set of points in parameter space where the true system exists such that the proximity measures for two adjacent filters are equivalent. As an example, for two adjacent filters k and k+1, this condition is expressed as

$$\ell_k = \ell_{k+1} \tag{4.68}$$

Without consideration of the variance of the proximity measure, the boundary has been previously considered a transition point from one filter to another. It is clear from Equation (4.67), that the transition from one filter to the next occurs over a region.

For example, consider two adjacent filters as shown in Figure 4.6. The proximity measure and one-sigma bounds are computed across the parameter space for each filter. The filter selected for an MMAE or MMAC is determined by $\min(\ell_k, \ell_{k+1})$ over the whole parameter space. The typical MMAE and MMAC prediction algorithms only consider the boundary to be where $\ell_k = \ell_{k+1}$. In the region designated $\ell = \ell_k$, it is clear

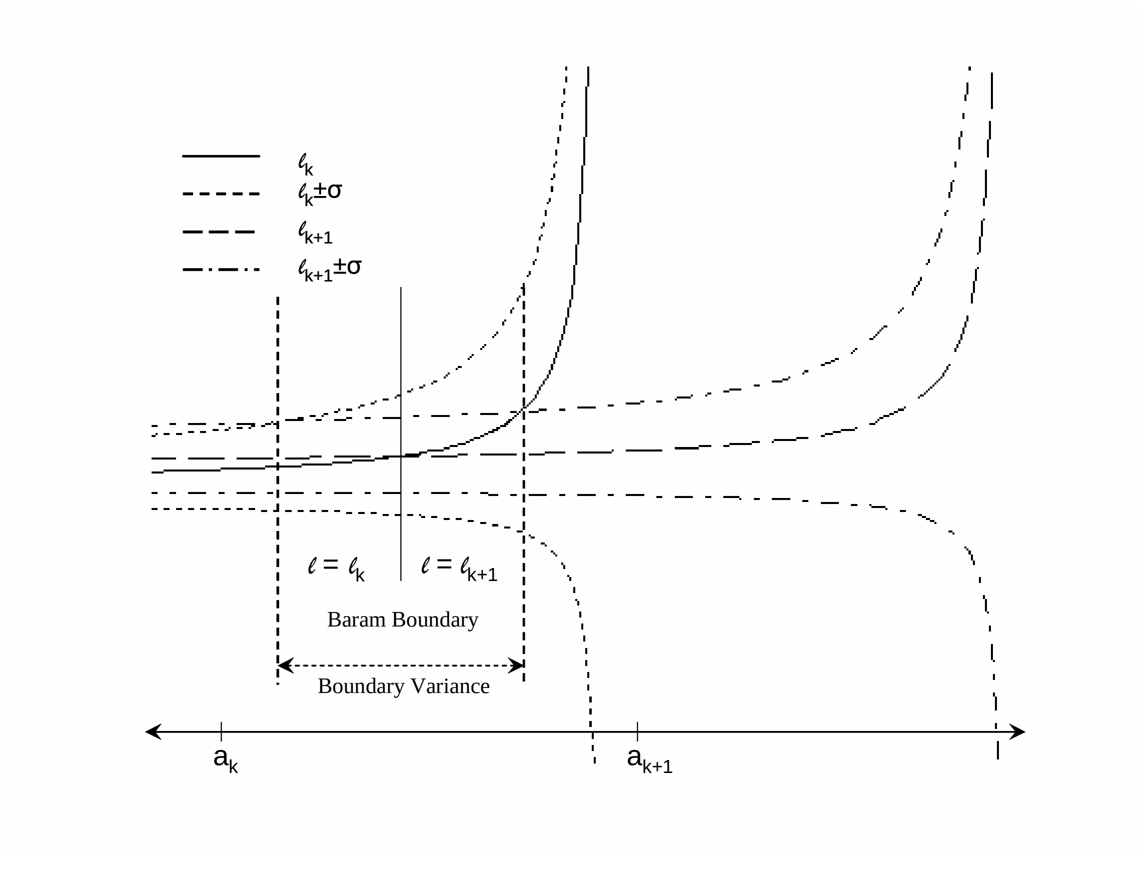

Figure 4.6 Proximity measure of two adjacent filters with one-sigma bounds.

that $\ell_{\mathrm{k}} < \ell_{\mathrm{k+1}}$ and it follows that $\ell_{\mathrm{k}} < \ell_{\mathrm{k+1}} + \sigma_{\ell_{k+1}}$ and $\ell_{\mathrm{k}} - \sigma_{\ell_\ell} < \ell_{\mathrm{k+1}}$. However, this example shows the possibility that, near the boundary, there are cases in which $\ell_{\mathrm{k}} + \sigma_{\ell_\ell} > \ell_{\mathrm{k+1}}$ and $\ell_{\mathrm{k}} + \sigma_{\ell_\ell} > \ell_{\mathrm{k+1}} + \sigma_{\ell_{\ell_{k+1}}}$. Now consider the region designated $\ell = \ell_{\mathrm{k+1}}$. There is the possibility that $\ell_{\mathrm{k+1}} + \sigma_{\ell_{k+1}} > \ell_{\mathrm{k}}$. These cases demonstrate the potential ambiguity in selecting a filter. Given that there is a spread in the possible selection of the filter, the boundary $\ell_{\mathrm{k}} = \ell_{\mathrm{k+1}}$ may not be conservative enough for the prediction algorithm that is used in parameter placement.

A more conservative optimization algorithm can protect against the ambiguity in the Baram boundary due to the sigma bounds on the proximity measure. In order to make the optimization algorithm more conservative, a larger boundary determined by the sigma bound for each proximity can be used. The example shows a one-sigma bound where some of the boundaries do not cross to form a potential crossover point. The proximity measure and sigma boundaries are system-dependent and thus it will be up to the designer on how conservative to make the boundary determination.

4.6 Summary

In this chapter the MMAC architecture is developed further and new approaches to design are presented. The investigation is based on the discovered enhancements for the LQG controllers considered in Chapter 3. Since the MMAC is essentially comprised of LQG controllers, the work is easily extended and adapted. The first extension is the enhancement to the point designs for the LQG. Improved performance of the individual LQG components is applied to improve the performance of the MMAC based upon those components. The second application of the modifications to the LQG is the enhanced robustness. In this case, the goal is to be robust to the limitations of the MMAC, namely, the limited number of LQG controllers in the MMAC. Finally, in the GMMAC architecture, a separate LQG controller fed by the sensor measurements replaces the single controller gain matrix. This is an actual change to the architecture, rather than just a design change to the LQG components. This approach allows for the blending of computed control independent of the filters that are used to determine the control weights. For all three modifications to the MMAC, the performance evaluations and optimal discretization algorithms are developed.

Two additional areas of research have been presented. First, a minor change to the Sheldon discretization algorithm is discussed. The change merely took advantage of the fact that the proximity measure can be calculated from the closed-loop state autocorrelation computation and the output autocorrelation can be obtained from the same calculation. This approach is more accurate than that suggested by Sheldon and saves computation time during discretization. The second area of improvement was the computation of the variance of the Baram distance measure. The transition from one filter that covers a portion of the parameter space to the next is a boundary that has a variation. The mean together with the variance of the Baram distance measure predicts the portion of parameter space over which the transition from one filter to the next occurs.

Clearly, the MMAC approach blends the output of the LQG controllers according to the computed conditional hypothesis weighting. The next chapter investigates MMAE-based control, in which the state estimate is blended and then multiplied by a gain to form the control. The research for the MMAE-based control contributed to the

MMAC approaches developed in this chapter. It should seem logical that the two approaches, though having the aforementioned basic difference, are closely related in some respects. This will be seen more clearly in the next chapter.

Chapter 5 - MMAE Based Control Development

This chapter discusses the research accomplished to advance the design and evaluation of the MMAE-based control architecture. As stated in Chapter 3, the original intention of the current research was to develop and demonstrate improvements to MMAE-based control by modifying the single full-state feedback controller in that architecture. Therefore, the design and performance improvements for the typical LQG controller discovered along the way and demonstrated in Chapter 3 will be used to improve the MMAE-based control. However, the full-state feedback control element is just one aspect of the MMAE-based control development.

Another area to be investigated is the analysis of various possible configurations of the MMAE-based architecture. Two general forms of architecture with several variations are considered. The first form investigated is the typical implementation characterized as using state estimates from an MMAE to feed a controller gain. A second configuration, alluded to in Chapter 2, is structured as an LQG controller *designed on the fly* based on parameter estimates from an MMAE. The analyses of these architectures are developed and comparisons with the MMAC are discussed.

As with the MMAC design, each of the stated approaches to MMAE-based control requires a method of discretization of the parameter space. Currently, an optimization-based method of discretization does not exist for any of the approaches to MMAE-based control. As will be discussed, discretization for best state estimate or parameter estimate in the MMAE portion alone does not provide a best solution for MMAE-based control. This final area of research presents discretization approaches for each of the proposed architectures that are derived from the techniques developed for the MMAC.

5.1 MMAE-Based Control Architecture Development

This section establishes the MMAE-based control architectures that will be examined in subsequent sections. Novel architectures and modification of typical approaches are considered. Typical of MMAE-based control is that each design approach is based on an MMAE used as a state and/or parameter estimator in tandem with a control element of some form. However, unique aspects for the proposed approaches are the manner in which the information from the MMAE portion is used and the form of the control portion.

Consider the MMAE-based control elements as presented in Chapter 2. The MMAE portion takes measurement inputs and produces state and parameter estimates. These state and parameter estimates are formed using the probability vector of weights associated with the elemental filters. This vector is not normally used other than to form the estimates. Now consider the control element component of the MMAE-based control. The typical control element takes a state estimate from the MMAE and multiplies by a gain to obtain the control. The control element is conventionally determined by the parameter estimate. Modifications to the typical design or selection of the control element and the form of the control element itself, not restricted to a full-state feedback gain matrix, are subjects of investigation in the sequel.

5.1.1 Control Element Selection

The concept of *selecting a controller* refers to using some table look-up type scheme to determine the control to apply. The *design on the fly* approach actually performs a design of the controller element in real time. Each of these two approaches has its advantages and disadvantages.

The advantages and disadvantages of *design on the fly* are related to performance and implementation, respectively. Clearly, a controller implemented with any synthesis technique using actual design parameters will outperform any approximation of the controller using the same design techniques or any synthesis based on approximations of design parameters. Of course, the obvious disadvantage is the computation time and resources required to perform the synthesis online. The synthesis must be accomplished faster than any changes in system parameters upon which the design is based. For a sampled data system, the control should be updated before the subsequent sample.

The advantages and disadvantages for the table look-up approach to specify the control are diametrically opposite to those for the *design on the fly* approach. Therefore, time and computational resource conservation are the main benefits. It takes very little time and computational resources to select the control from a stored table. Clearly, approximation of control is the main disadvantage. There is only a limited number of discrete models that represent the system on which to perform control synthesis. The more controllers that populate the look-up table, the closer the approximation will be to the true system. Another aspect to selecting the control can be categorized according to what information is used and how is it used to obtain the approximated controller from the look-up table. Typically, estimates of the system parameters are used as indices in the table. As previously mentioned, the probability vector that is used in forming the parameter estimate is additional information that may be considered as an alternative or an augmentation to the parameter estimate itself (i.e., the spread of the density function for parameters may be useful, as well as the density's center of mass).

This dissertation investigates the table look-up approach for the selection of a control element. In practicality, the implementation of MMAE-based control will be driven to this approach because of computation time and resources. Also, *design on the fly* will be the limiting best case for any approximations based on parameter estimates. Thus, of course, architectures based on table look-up using a discretization of the parameter space and being fed by an explicit parameter estimate is one of the possible architectures for further study and development. Two other architectures for study are based on the additional information given by the vector of probabilities for the filters.

Consider the potential advantage of using the vector of probabilities associated with the filters in the control selection scheme. Clearly, the state estimate and parameter estimate are both based on the probability weighting of the possible filter models. Using the parameter estimate to select the control can be problematic when considering that a single estimate can be derived from more than one probability weight distribution. For example, given three filters placed at arbitrary values of A_1, A_2, and A_3 and the parameter estimate is A. Then the parameter estimate is determined by

$$\hat{A} = p_1 A_1 + p_2 A_2 + p_3 A_3 \tag{5.1}$$

Eliminating one of the probabilities and rewriting in terms of the other two yields

$$\hat{A} = (1 - p_2 - p_3) A_1 + p_2 A_2 + p_3 A_3 \tag{5.2}$$

$$p_3 = (\hat{A} + p_2 A_1 - p_2 A_2)/(A_3 - A_1) \tag{5.3}$$

This coupled with the fact that $p_1 = 1 - p_2 - p_3$, yields the result that a *nonunique* probability vector can be associated with any specified parameter estimate $\hat{A}$, given an arbitrary location of filters A_1, A_2, and A_3. For example, if A_1=10, A_2=20, and A_3=30 and the returned parameter estimate is $\hat{A}$ = 25, then Table 5.1 gives some possible probability values. It is clear from the entries in the table that one control law for a parameter estimate may not be adequate, since different control strategies might be optimal for the various possibilities in Table 5.1, yet they correspond to the same $\hat{A}$.

p_1	p_2	p_3
0.15	0.20	0.65
0.10	0.30	0.60
0.05	0.40	0.55
0.00	0.50	0.50
0.05	0.60	0.45

Table 5.1 Possible probabilities for parameter estimate example

Consider two methods for using the unique information provided by the probability vector. The first such method is not exactly a table look-up approach, but rather uses the probabilities as weights. The second method uses the elements of the probability vector as an index into a table of predesigned controllers.

Using the probabilities as weights on a set of predesigned controllers requires one controller design for each element of the probability vector. Since the elements in the probability vector are associated with a filter, the controller is designed to correspond to that filter. The control is determined by:

$$\mathbf{G}_{\mathrm{c}}^{*}(\mathbf{p}) = \sum_{\mathrm{k}=1}^{\mathrm{N}} p_{\mathrm{k}} \mathbf{G}_{\mathrm{c}}^{*}(\mathbf{a}_{\mathrm{c\,k}}) \tag{5.4}$$

A controller is designed for the parameter value $\mathbf{a}_{ck}$ for k = 1…N, where N is the number of filters. The parameter value corresponds to a filter, but it is not necessarily the same as the filter parameter and is thus denoted as $\mathbf{a}_c$.

Note that in Equation (5.4), the controllers at each discrete point are fixed. Thus, the control that is applied to the state estimate from the MMAE is a function of the probabilities. In Equation (5.4) the probabilities are used to blend the predesigned controllers. Clearly, this is different from using the parameter estimate derived from the probabilities to select the control from a table. This approach is very similar to that used for the MMAC, but for generating $\mathbf{u}$ versus $\mathbf{G}_c^*$ and $\hat{\mathbf{x}}$ separately. Blending preserves the uniqueness of the information from the probabilities rather than it being obscured in it in the parameter estimate.

The second approach that uses the probability information is a table look-up using the elements of the probability vector as indices. The number of indices or dimensions of the table is equivalent to the number of filters. Each index maps the possible range of the probability for the filter, i.e., in the range of 0 to 1. The number of elements for each index determines the discrete values of the probability. Accordingly, the size of the table will be the number elements for each index times the number of indices (which is the number of filters). The controller in the look-up table is optimized for the probability vector values that correspond to the indices. The probability vector used for the optimization is also the same probability vector that forms the state estimate. This approach provides a unique mapping from the probability used to derive the state estimate to the corresponding control. This mapping is not unique for the parameter-estimate-based table look-up approach. As demonstrated previously, a single parameter estimate can be derived from more than one probability vector and thus the mapping is not unique. Both of these approaches will be further developed in Section 5.3.

5.1.2 Type of Control Element

The second major attribute in the MMAE-based control architecture to consider for modification is the controller element. There are three different approaches to control considered in this research. Subsequent sections present the optimized design for each of these proposed architecture implementations.

The first approach to control is to use an MMAE to obtain a state estimate that is fed to a full-state feedback controller gain matrix. The controller gain is non-adaptive and typically designed for some nominal plant. The only adaptive element in this architecture is the state estimator. Thus, as will be discussed and demonstrated, this control approach is only as effective as the robustness of the feedback controller, which is specified by its gain matrix.

A second approach alluded to in the previous section and the one that is most typical, also uses adaptive state estimation, and in addition, uses an adaptive selection scheme to determine a full-state feedback controller gain as well. The typical approach assumes that the parameter estimate accurately describes the plant to be controlled, and that passing such a parameter estimate to the feedback gain computation process is sufficient for adaptation purposes. Thus, there is adaptation of the control generation process as well as the state estimation process. However, as discussed in Chapter 2, there is an inherent trade-off between obtaining accurate state and parameter estimates. As will be discussed in the sequel, this trade-off does not exist for optimized table look-up schemes.

A third and novel approach to MMAE-based control is to use an MMAE to obtain a parameter estimate, which is then used to select a single LQG controller. This is different from the previous two approaches since the MMAE-generated state estimate is not used and the control element is a single full-state feedback controller. This approach does not suffer from the trade-off between parameter and state estimation since the MMAE portion needs only to be designed for parameter estimation. As with the previous control approaches, the LQG determination can be accomplished with a selection scheme. Obviously, more information has to be stored in the look-up table than just a gain matrix.

The three proposed control element implementations along with the selection schemes are all similar to the MMAC in some respects. It should be apparent from the subsequent MMAC and MMAE-based control comparison discussions that the full benefits of any of the proposed approaches may only be obtained by continually forced blending, perhaps by a moving-bank type method [37,44, 63,64,65]. This is due to the fact that the proposed architectures are all dependent on the MMAE to provide the state and parameter estimates that will reach a steady state value corresponding to the filter locations.

5.2 MMAE-Based Control In Comparison to MMAC

Relevant work in MMAE-based control was reviewed in Chapter 2 as a separate but similar approach as MMAC to adaptive control. The resemblance between the two implementations is worth investigating in order to lay the foundation for the MMAE-based controller design discussion. This section utilizes the typical implementation of

MMAE-based control: Control element selected based on parameter estimate information fed by the state estimate from the MMAE.

Though extensively developed in Chapter 4, for comparison with MMAE-based control, first reconsider the MMAC approach to adaptive control. Each elemental controller output is calculated from the state estimate output of the filter and the controller based on the parameter $\mathbf{a}_{k}$.:

$$\mathbf{u}_{k} = -\mathbf{G}_{c}^{*}(\mathbf{a}_{k})\hat{\mathbf{x}}_{k} \tag{5.5}$$

The control is computed as the probability-weighted average of the $\mathbf{u}_{k}$'s:

$$\mathbf{u} = \sum_{k=1}^{N} p_{k}\mathbf{u}_{k} \tag{5.6}$$

Thus, in terms of the probabilities, state estimates and controller gains, the controller output is given by:

$$\mathbf{u} = -p_{1}\mathbf{G}_{c}^{*}(\mathbf{a}_{1})\hat{\mathbf{x}}_{1} - p_{2}\mathbf{G}_{c}^{*}(\mathbf{a}_{2})\hat{\mathbf{x}}_{2} \cdots - p_{N}\mathbf{G}_{c}^{*}(\mathbf{a}_{N})\hat{\mathbf{x}}_{N} \tag{5.7}$$

Now consider MMAE-based control in which the controller is evaluated based on the parameter estimate $\hat{\mathbf{a}}$. This gain is multiplied by the state estimate to yield the output. Thus, the control output is calculated as:

$$\mathbf{u} = -\mathbf{G}_{c}^{*}(\hat{\mathbf{a}})\hat{\mathbf{x}} \tag{5.8}$$

With the state estimate calculation expanded by:

$$\hat{\mathbf{x}} = \sum_{k=1}^{N} p_{k}\hat{\mathbf{x}}_{k} \ , \tag{5.9}$$

the control output becomes:

$$\begin{aligned} \mathbf{u} &= -\mathbf{G}_{c}^{*}(\hat{\mathbf{a}})(p_{1}\hat{\mathbf{x}}_{1} + p_{2}\hat{\mathbf{x}}_{2} \cdots p_{N}\hat{\mathbf{x}}_{N}) \\ &= -p_{1}\mathbf{G}_{c}^{*}(\hat{\mathbf{a}})\hat{\mathbf{x}}_{1} - p_{2}\mathbf{G}_{c}^{*}(\hat{\mathbf{a}})\hat{\mathbf{x}}_{2} \cdots - p_{N}\mathbf{G}_{c}^{*}(\hat{\mathbf{a}})\hat{\mathbf{x}}_{N} \end{aligned} \tag{5.10}$$

The controller gain is evaluated according to:

$$\mathbf{G}_c^*(\hat{\mathbf{a}}) = w_1\mathbf{G}_c^*(\mathbf{a}_{c1}) + w_2\mathbf{G}_c^*(\mathbf{a}_{c2}) \cdots + w_M\mathbf{G}_c^*(\mathbf{a}_{cM}) \tag{5.11}$$

where $\mathbf{G}_c^*(\mathbf{a}_{ck})$ is the point controller designed based on $\mathbf{a}_{ck}$, the kth discrete point in the discretized uncertain parameter space. The weighting w_i is determined by the controller evaluation algorithm or table look-up. Now, substitution of the controller evaluation Equation (5.11) into Equation (5.10) yields the full expression for the control computation:

$$\begin{aligned}\mathbf{u} = &-p_1\left(w_1\mathbf{G}_c^*(\mathbf{a}_{c1}) + w_2\mathbf{G}_c^*(\mathbf{a}_{c2}) + \cdots + w_M\mathbf{G}_c^*(\mathbf{a}_{cM})\right)\hat{\mathbf{x}}_1 \\ &-p_2\left(w_1\mathbf{G}_c^*(\mathbf{a}_{c1}) + w_2\mathbf{G}_c^*(\mathbf{a}_{c2}) + \cdots + w_M\mathbf{G}_c^*(\mathbf{a}_{cM})\right)\hat{\mathbf{x}}_2 \\ &\qquad\vdots \\ &-p_N\left(w_1\mathbf{G}_c^*(\mathbf{a}_{c1}) + w_2\mathbf{G}_c^*(\mathbf{a}_{c2}) + \cdots + w_M\mathbf{G}_c^*(\mathbf{a}_{cM})\right)\hat{\mathbf{x}}_N\end{aligned} \tag{5.12}$$

Note that the control table can have a larger number of entries than parameter estimators and the weighting of entries does not have to be probability based, as was the case for MMAC. In fact, the weighting between entries could be some nearest neighbor based weighting which would allow zero weighting for most entries in the table. In addition, by allowing a finer discretization on the control gain table, the interpolated gain value based on a derived $\hat{\mathbf{a}}$, will approach the controller gain value designed for that given $\hat{\mathbf{a}}$. The *design on-the-fly* or controller look-up function as part of the MMAE-based controller will be covered in more details in Sections 5.3.3.

With the two expressions for the MMAC and the MMAE-based control derived in Equations (5.7) and (5.12), direct similarities between the two methods can be discerned. In order to bring the form of the MMAE-based control structure closer to that of the MMAC, certain conditions are required. First, it is assumed that the MMAE-based control is implemented with a table look-up approach. Second, it is assumed that the controllers in the gain look-up table are designed, based on the same parameter locations as the controller gains for the MMAC. This obviously requires the number of gains in the table look-up and number of elemental filters for the MMAC to be the same. In addition, the discretization of the parameter space for the MMAC is the same as for the parameter space for the MMAE portion of the MMAE-based controller. Finally, the interpolation between the entries in the controller gain table has to be equivalent to the estimation calculation based on probability weighting. This condition is equivalent to setting

$$\mathrm{w}_{\mathrm{k}} = p_{\mathrm{k}} \quad \forall\, \mathrm{k}$$

in Equation (5.12). Thus, the gain table look-up is probability-based and the controller calculation is expressed as:

$$\mathbf{G}_{\mathrm{c}}^{*}(\hat{\mathbf{a}}) = \sum_{\mathrm{k}=1}^{\mathrm{N}} p_{\mathrm{k}} \mathbf{G}_{\mathrm{c}}^{*}(\mathbf{a}_{\mathrm{c\,k}}) \tag{5.13}$$

The evaluation of the controller becomes a function of this expanded parameter estimation and the controller output is accordingly expressed as:

$$\mathbf{u} = -[p_1 \mathbf{G}_{\mathrm{c}}^{*}(\mathbf{a}_{\mathrm{c}1}) + p_2 \mathbf{G}_{\mathrm{c}}^{*}(\mathbf{a}_{\mathrm{c}2}) \cdots p_{\mathrm{N}} \mathbf{G}_{\mathrm{c}}^{*}(\mathbf{a}_{\mathrm{c\,N}})](p_1 \hat{\mathbf{x}}_1 + p_2 \hat{\mathbf{x}}_2 \cdots p_{\mathrm{N}} \hat{\mathbf{x}}_{\mathrm{N}}) \tag{5.14}$$

This expression can be expanded to yield a special case of the equation for MMAE-based control:

$$\begin{aligned} \mathbf{u} = &-p_1\Big(p_1 \mathbf{G}_{\mathrm{c}}^{*}(\mathbf{a}_{\mathrm{c}1}) + p_2 \mathbf{G}_{\mathrm{c}}^{*}(\mathbf{a}_{\mathrm{c}2}) + \cdots + p_{\mathrm{N}} \mathbf{G}_{\mathrm{c}}^{*}(\mathbf{a}_{\mathrm{cN}})\Big)\hat{\mathbf{x}}_1 \\ &- p_2\Big(p_1 \mathbf{G}_{\mathrm{c}}^{*}(\mathbf{a}_{\mathrm{c}1}) + p_2 \mathbf{G}_{\mathrm{c}}^{*}(\mathbf{a}_{\mathrm{c}2}) + \cdots + p_{\mathrm{N}} \mathbf{G}_{\mathrm{c}}^{*}(\mathbf{a}_{\mathrm{cN}})\Big)\hat{\mathbf{x}}_2 \\ &\qquad\vdots \\ &- p_{\mathrm{N}}\Big(p_1 \mathbf{G}_{\mathrm{c}}^{*}(\mathbf{a}_{\mathrm{c}1}) + p_2 \mathbf{G}_{\mathrm{c}}^{*}(\mathbf{a}_{\mathrm{c}2}) + \cdots + p_{\mathrm{N}} \mathbf{G}_{\mathrm{c}}^{*}(\mathbf{a}_{\mathrm{cN}})\Big)\hat{\mathbf{x}}_{\mathrm{N}} \end{aligned} \tag{5.15}$$

Clearly, the MMAC and the MMAE-based controllers are *not* equivalent. The controller gains for the MMAC correspond to the specific state estimates. For the

MMAE-based controller, each state estimate is multiplied by a gain based on probability-based blending of all possible point designs in the controller table. The differences are illustrated more clearly by rewriting the controller equation in terms of vector matrix notation. The MMAC is expressed as

$$\mathbf{u} = -\left[\mathbf{G}_c^*(\mathbf{a}_1) \quad \mathbf{G}_c^*(\mathbf{a}_2)\ldots\mathbf{G}_c^*(\mathbf{a}_N)\right]\left[diag(p_1\mathbf{I}, p_2\mathbf{I}, \cdots p_N\mathbf{I})\right]\begin{bmatrix}\hat{\mathbf{x}}_1 \\ \hat{\mathbf{x}}_2 \\ \vdots \\ \hat{\mathbf{x}}_N\end{bmatrix} \tag{5.16}$$

and the equation for MMAE-based control is

$$\mathbf{u} = -\left[\mathbf{G}_c^*(\mathbf{a}_{c1}) \quad \mathbf{G}_c^*(\mathbf{a}_{c2})\ldots\mathbf{G}_c^*(\mathbf{a}_{cN})\right]\begin{bmatrix} p_1^2\mathbf{I} & p_1p_2\mathbf{I} & \cdots & p_1p_N\mathbf{I} \\ p_2p_1\mathbf{I} & p_2^2\mathbf{I} & \cdots & p_2p_N\mathbf{I} \\ & & \vdots & \\ p_Np_1\mathbf{I} & p_Np_2\mathbf{I} & \cdots & p_N^2\mathbf{I}\end{bmatrix}\begin{bmatrix}\hat{\mathbf{x}}_1 \\ \hat{\mathbf{x}}_2 \\ \vdots \\ \hat{\mathbf{x}}_N\end{bmatrix} \tag{5.17}$$

The inner matrix in Equation (5.16) and in Equation (5.17), is based on the probabilities which clearly show that there are cross terms for the MMAE-based control not present in MMAC. These cross terms cannot be considered negligible. Take the case in which there is a group of probabilities that are close and obviously not near one; the off-diagonal terms will be nearly equivalent to the diagonal term. Also, consider that the diagonal terms are the square of the diagonal terms of the MMAC. Obviously, since probabilities are always less than one, the p term with the largest probability will be reduced by the squaring operation. In fact, in Equation (5.17), adding the jth column of blocks in the second matrix on the right-hand side clearly yields $p_j\mathbf{I}$, as would adding the jth row of blocks.

As presented in Chapter 4, the optimization of the MMAC only considers the steady state point to evaluate the controlled system. At steady state, it is assumed that the single filter has a probability of one. As illustrated in Figure 5.1, the MMAC architecture reduces to single filter that feeds a control gain matrix and the probability multiplier is one. All other filters and associated gains are no longer part of the MMAC at steady state. In this case, the output in Equation (5.16) for, say the jth filter, becomes:

$$\mathbf{u} = -\left[\mathbf{G}_c^*(\mathbf{a}_1) \quad \mathbf{G}_c^*(\mathbf{a}_2)\ldots\mathbf{G}_c^*(\mathbf{a}_j)\ldots\mathbf{G}_c^*(\mathbf{a}_N)\right]\left[\mathrm{diag}(\mathbf{0},\mathbf{0},\cdots\mathbf{I}\cdots\mathbf{0})\right]\begin{bmatrix}\hat{\mathbf{x}}_1\\ \hat{\mathbf{x}}_2\\ \vdots\\ \hat{\mathbf{x}}_j\\ \vdots\\ \hat{\mathbf{x}}_N\end{bmatrix} \tag{5.18}$$

which reduces to

$$\mathbf{u} = -\mathbf{G}_c^*(\mathbf{a}_j)\hat{\mathbf{x}}_j \tag{5.19}$$

Now consider the MMAE-based control and take a similar approach to evaluate the performance at steady state. As with the MMAC, one filter in the MMAE portion of the controller will assume all the probability at steady state. As shown in Figure 5.2, as was just discussed to be the case with the MMAC, the architecture reduces to a single

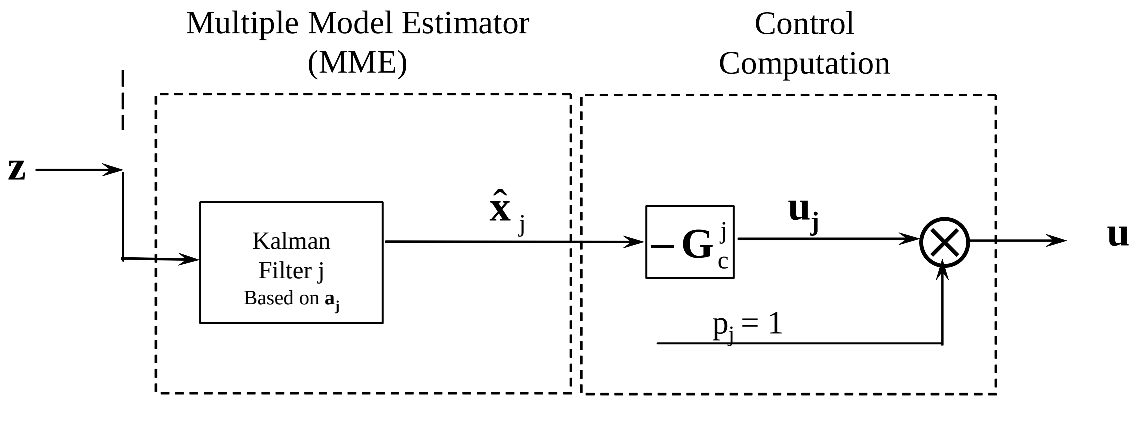

Figure 5.1 MMAC control computation at steady state

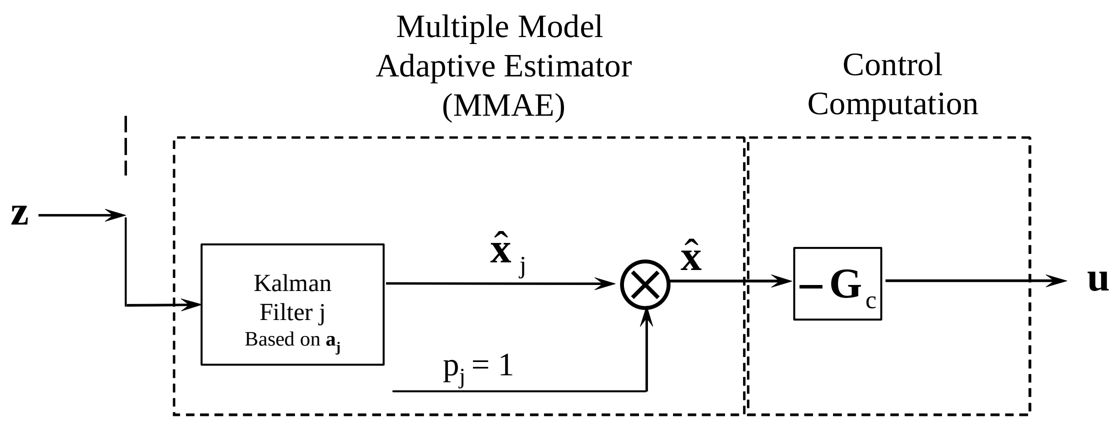

Figure 5.2 MMAE-based control at steady state.

filter that feeds a control gain matrix. Again, consider that the jth filter has assumed all the probability at steady state and Equation (5.17) becomes:

$$\mathbf{u} = -\left[\mathbf{G}_c^*(\mathbf{a}_{c1}) \quad \mathbf{G}_c^*(\mathbf{a}_{c2}) \ldots \mathbf{G}_c^*(\mathbf{a}_{cj}) \ldots \mathbf{G}_c^*(\mathbf{a}_{cN})\right] \begin{bmatrix} \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{x}}_1 \\ \hat{\mathbf{x}}_2 \\ \vdots \\ \hat{\mathbf{x}}_j \\ \vdots \\ \hat{\mathbf{x}}_N \end{bmatrix} \tag{5.20}$$

which reduces to

$$\mathbf{u} = -\mathbf{G}_c^*(\mathbf{a}_{cj})\hat{\mathbf{x}}_j \tag{5.21}$$

Clearly from Equations (5.19) and (5.21), it is evident the control for MMAC and MMAE-based control are identical at steady state. Specifically, the two different control architectures produce the same output when a single filter in the respective architectures has assumed a probability of one. Further, regardless of the type of feedback gain look-up scheme implemented for the MMAE-based controller, control at steady state reduces to a state estimate that feeds a control gain. Therefore, any performance improvements of the MMAE-based control over the MMAC must only be realized in the transient since steady state control will be identical.

Now consider the output for the two control approaches during the transient phase. From Equation (5.16), it is evident that the MMAC performs a linear blending of the control according to the probabilities until steady state is reached. For the MMAE-based control as proposed in this section, the control is non-linear in the probabilities assigned to the filters and really should not be considered a table look-up selection of the controller. Of course, the MMAC approach is not a table look-up of full-state feedback control gains either. However, Section 5.3.3 will present a different approach for determining the feedback gain that *will* be based on a table look-up scheme.

5.3 MMAE-Based Control Evaluation and Design

This section develops design procedures for the architecture modifications proposed in the previous sections. These procedures are very similar to the methods for the MMAC as discussed in Chapter 4, but adapted for MMAE-based control. As was the case for MMAC, it is first necessary to develop a performance evaluation measure. In order to maintain consistency for comparisons with the MMAC, the performance criterion will be the steady state position autocorrelation. The previous section established the fact that the form of the evaluation equations will be very similar to the equations for the MMAC.

With the performance evaluation equations established, subsequent discussions define the procedures for designing the components of the architecture. MMAC design is concerned with discretization of the parameter space. The newly developed techniques for discretization in Chapter 4 utilize a given a set of predesigned filters and corresponding controllers. Again, from the discussions in the previous section, it should be apparent that an approach similar to that used for the MMAC will determine where the filters reside in parameter space for the MMAE-based control. Thus, the remaining design element is the controller portion of the proposed architectures. Controllers must be designed for each possible point in parameter space that may be indicated by the controller selection algorithm.

Each discussion of the proposed MMAE-based control architecture will conclude with analysis of projected advantages and disadvantages of each scheme and implementation issues. Of course, any advantage or disadvantage may be viewed differently when one implementation is compared to another. So, where appropriate, the proposed architecture is compared to both the MMAC and other MMAE-based control implementations. The analysis provides a general assessment of results for the example problem discussed in Chapter 6.

5.3.1 MMAE-Based Control with Nominal Controller Element

This basic MMAE-based control architecture has a nominal controller as the only possible control element. Hence, there is not a controller selection scheme. It is the simplest architecture to implement in terms of computation requirements. Note that this structure is a special case of the MMAE-based control architecture introduced in Chapter 2. As shown Figure 5.3, rather than using the parameter estimate to select a full-state feedback controller gain, the state estimate is fed to a *single* controller gain based on some *nominal* controller design.

5.3.1.1 Performance Evaluation Equations

As with the previous architectures, performance evaluation development begins with the derivation of the steady-state output correlation equations. Since this architecture builds

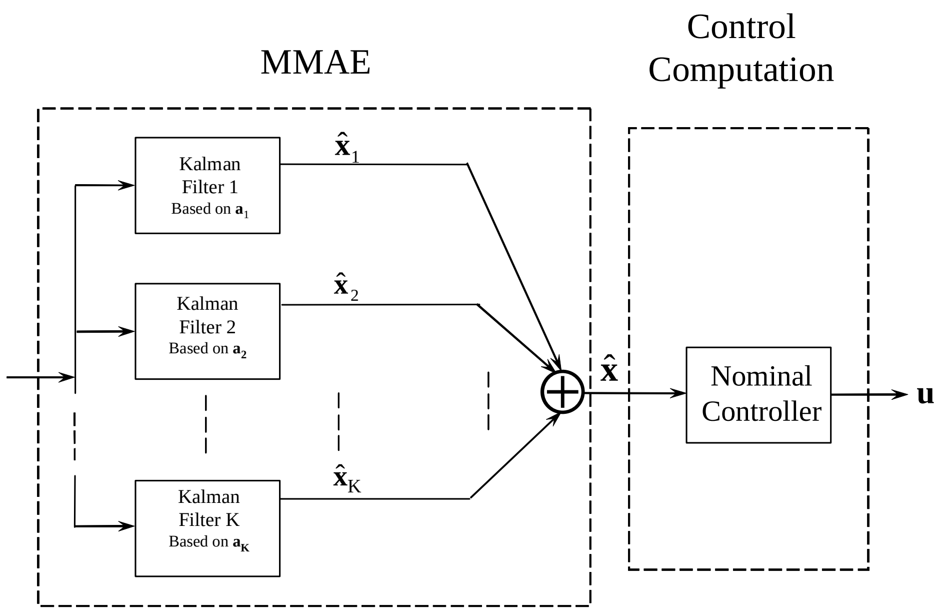

Figure 5.3 MMAE-based control with a nominal controller

upon the MMAE similarly to the MMAC, the derivation will follow the same procedure. Likewise, the equations will be similar to those in Chapter 2 for the MMAC.

The derivation begins with the basic equation for the control. With a controller gain matrix that is designed based on a nominal plant in the space of possible parameters that describes the possible plants, the control is expressed as:

$$\mathbf{u}(t_i) = -\mathbf{G}_c^*(t_i, \mathbf{a}_{\text{nom}})\hat{\mathbf{x}}_{\text{MMAE}}(t_i^+) \tag{5.22}$$

The state estimate from the MMAE can be expanded to yield:

$$\begin{aligned}\mathbf{u}(t_i) &= -\mathbf{G}_c^*(t_i, \mathbf{a}_{\text{nom}})\left\{p_1\hat{\mathbf{x}}_1(t_i^+) + p_2\hat{\mathbf{x}}_2(t_i^+)\cdots + p_N\hat{\mathbf{x}}_N(t_i^+)\right\} \\ &= -p_1\mathbf{G}_c^*(t_i, \mathbf{a}_{\text{nom}})\hat{\mathbf{x}}_1(t_i^+) - p_2\mathbf{G}_c^*(t_i, \mathbf{a}_{\text{nom}})\hat{\mathbf{x}}_2(t_i^+)\cdots - p_N\mathbf{G}_c^*(t_i, \mathbf{a}_{\text{nom}})\hat{\mathbf{x}}_N(t_i^+)\end{aligned} \tag{5.23}$$

Now denote the gain matrix as:

$$\mathbf{G}_{c\,\text{nom}}^* = \mathbf{G}_c^*(t_i, \mathbf{a}_{\text{nom}}) \tag{5.24}$$

which simplifies Equation (5.23) as:

$$\mathbf{u}(t_i) = -p_1\mathbf{G}^*_{\mathrm{c\,nom}}\hat{\mathbf{x}}_1(t_i^+) - p_2\mathbf{G}^*_{\mathrm{c\,nom}}\hat{\mathbf{x}}_2(t_i^+)\cdots - p_{\mathrm{N}}\mathbf{G}^*_{\mathrm{c\,nom}}\hat{\mathbf{x}}_{\mathrm{N}}(t_i^+) \tag{5.25}$$

Now, as was the case for MMAC [56,57], assume that at steady state, one of the filters, say the k$^{\mathrm{th}}$, will have a probability of one. Equation (5.25) now reduces to:

$$\mathbf{u}(t_i) = -\mathbf{G}^*_{\mathrm{c\,nom}}\,\hat{\mathbf{x}}_{\mathrm{k}}(t_i^+) \tag{5.26}$$

Except for the controller gain matrix, the control in Equation (5.26) is identical in form to the MMAC. Thus, the performance equation derivation will follow that for the MMAC. Thus, without repeating the derivation from Chapter 2 and now using the notation for the nominal controller gain matrix, the state equations are:

$$\begin{aligned}\begin{bmatrix}\mathbf{x}_{\mathrm{t}}(t_{i+1})\\ \hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^-)\end{bmatrix} &= \begin{bmatrix}(\mathbf{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c\,nom}}\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}}) & -\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c\,nom}}(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\\ (\mathbf{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^*_{\mathrm{c\,nom}})\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}} & (\mathbf{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^*_{\mathrm{c\,nom}})(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\end{bmatrix}\begin{bmatrix}\mathbf{x}_{\mathrm{t}}(t_i)\\ \hat{\mathbf{x}}_{\mathrm{k}}(t_i^-)\end{bmatrix}\\ &+ \begin{bmatrix}-\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c\,nom}}\mathbf{K}_{\mathrm{k}}\\ (\mathbf{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^*_{\mathrm{c\,nom}})\mathbf{K}_{\mathrm{k}}\end{bmatrix}\mathbf{v}_{\mathrm{t}}(t_i) + \begin{bmatrix}\mathbf{G}_{\mathrm{d\,t}}\\ \mathbf{0}\end{bmatrix}\mathbf{w}_{\mathrm{d\,t}}(t_i)\end{aligned} \tag{5.27}$$

Again, this assumes that one filter has assumed all the probability at steady state.

Now define

$$\mathbf{T}_{\mathrm{nom}} = \begin{bmatrix}(\mathbf{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c\,nom}}\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}}) & -\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c\,nom}}(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\\ (\mathbf{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^*_{\mathrm{c\,nom}})\mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{t}} & (\mathbf{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^*_{\mathrm{c\,nom}})(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}})\end{bmatrix} \tag{5.28}$$

$$\mathbf{L}_{\mathrm{nom}} = \begin{bmatrix}\mathbf{G}_{\mathrm{d\,t}} & -\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c\,nom}}\mathbf{K}_{\mathrm{k}}\\ \mathbf{0} & (\mathbf{\Phi}_{\mathrm{k}} - \mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^*_{\mathrm{c\,nom}})\mathbf{K}_{\mathrm{k}}\end{bmatrix} \tag{5.29}$$

and the statistics for the noise as:

$$E\left\{\begin{bmatrix}\mathbf{w}_{\mathrm{d\,t}}(t_i)\\ \mathbf{v}(t_i)\end{bmatrix}\begin{bmatrix}\mathbf{w}_{\mathrm{d\,t}}{}^{\mathrm{T}}(t_j) & \mathbf{v}^{\mathrm{T}}(t_j)\end{bmatrix}\right\} = \begin{cases}\mathbf{Q_0}(t_i) & t_i = t_j\\ \mathbf{0} & t_i \neq t_j\end{cases} \tag{5.30}$$

where

$$\mathbf{Q_0}(t_i) = \begin{bmatrix}\mathbf{Q}_{\mathrm{d\,t}}(t_i) & \mathbf{0}\\ \mathbf{0} & \mathbf{R}_{\mathrm{t}}(t_i)\end{bmatrix} \tag{5.31}$$

The output autocorrelation of the augmented system at time t_{i+1} can be written conveniently as:

$$\mathrm{E}\left\{\begin{bmatrix}\mathbf{x}_{\mathrm{t}}(t_{i+1})\\ \hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^{-})\end{bmatrix}\begin{bmatrix}\mathbf{x}_{\mathrm{t}}^{\mathrm{T}}(t_{i+1}) & \hat{\mathbf{x}}_{\mathrm{k}}^{\mathrm{T}}(t_{i+1}^{-})\end{bmatrix}\right\} = \boldsymbol{\Xi}_{\mathrm{k}}(t_{i+1}^{-}) = \mathbf{T}_{\mathrm{nom}}\,\boldsymbol{\Xi}_{\mathrm{k}}(t_{i}^{-})\,\mathbf{T}_{\mathrm{nom}}^{\mathrm{T}} + \mathbf{L}_{\mathrm{nom}}\mathbf{Q}_{\mathbf{0}}\mathbf{L}_{\mathrm{nom}}^{\mathrm{T}} \qquad (5.32)$$

This equation is identical to the expression for the MMAC. However, the main difference is that the controller gain now is not related directly to any particular filter. Thus, the controller may not correspond to the k$^{\text{th}}$ filter as specified in Equation (5.27), but to some other point in the parameter space. Accordingly, the evaluation of Equation (5.32) will differ from its MMAC counterpart. The evaluation is not just a function of the filter locations, but also the parameters that describe the controller. Hence, the discretization algorithm will be slightly different from the MMAC and it is expected that the resultant discretization of the parameter space will not be the same.

5.3.1.2 Discretization Algorithm

Although the MMAE-based control with a nominal controller is the simplest architecture, it is a little more complex to design than the MMAC. Now, not only is the optimal placement of filters to be determined, the optimal placement of the controller has to be specified as well. This is only a slightly more involved procedure than simply determining which is the optimal controller for a single filter location because there is not a one-to-one correspondence.

To determine the optimal placement of the filter, as with the MMAC, the cost function minimized is the output squared over the range of possible values of the parameters and is expressed as:

$$J_{2\,c} \equiv \frac{\int_A E\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y}\}da}{\int_A da} \tag{5.33}$$

where $\mathrm{E}\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\,\mathbf{y}\} \equiv \mathrm{tr}(\mathbf{W}\,\mathbf{C}\,\boldsymbol{\Psi}_{\mathbf{x}_t}\mathbf{C}^{\mathrm{T}})$ and $\boldsymbol{\Psi}_{\mathbf{x}_t} = \mathrm{E}\{[\mathbf{x}_t][\mathbf{x}_t]^{\mathrm{T}}\}$. Though Equation (5.33) is identical to the cost function from Chapter 2 for the MMAC discussion, in this case the cost is a function of the filter locations and nominal controller design point.

The discretization algorithm is very similar to the MMAC and the steps are summarized as follows:

1) Describe in terms of the parameter vector **a**, the truth model of the system, the filter and nominal controller.
2) Choose the number of filters K in the MMAE.
3) Choose a representative parameter set, $\mathcal{A} \equiv \{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_K, \mathbf{a}_{nom}\}$ to begin the minimization; $\mathbf{a}_{nom}$ specifies the parameter for the controller.
4) Use a numerical integration technique to evaluate $J_{2\,c}$. This evaluation depends on determining to which filter the MMAE-based control will converge and determining the nominal controller. Determination of the convergence is discussed in Section 2.5.3.
5) Use a vector minimization technique with the functional evaluation from Step (4) to minimize $J_{2\,c}$.

5.3.1.3 Probability Lower Bounding and Transient Response Evaluation

As noted in Chapter 2, in order to negate the effects of filter lock-out, the filters can be designed with an assumed artificial lower bound. To implement this lower bound, assume all the filters will have a probability of $p_{\min}$, except for one filter. That filter will have a probability expressed as:

$$p_{\text{sel}} = 1 - p_{\min}(K-1) \tag{5.34}$$

This of course can be incorporated into the design of the MMAE-control with a nominal controller. However, important to the MMAE-based control development is the ability to assess the performance in the transient phase.

Given that the steady state responses will be similar if not identical to MMAC, a measure of the performance of the transient phase will give an assessment the benefits of the MMAE-based control approach. Previous development in this section assumes that the system has reach steady state when one of the filters has the maximum probability. Developing a performance measure for implementing lower bounding for the MMAE-based control with a nominal controller provides the capability for an assessment of the transient phase, or in other words, before one of the filters has assumed the maximum probability.

The derivation of the autocorrelation equations for implementing lower bounding borrows the results from the MMAC development. The control expressed in Equation (5.25) is of the same form as the MMAC. However, the main difference is that the controller gain $\mathbf{G}^{*}_{c_{\text{nom}}}$ is applied to each state estimate rather than a different gain for each state estimate. Thus, the performance measure equations will be similar to the MMAC. Each controller gain in the MMAC expression is simply replaced with the single nominal controller gain. The state equations with lower bounding are given without further development and are expressed as:

$$
\begin{bmatrix} \mathbf{x}_{\mathrm{t}}(t_{i+1}) \\ \hat{\mathbf{x}}_1(t_{i+1}^-) \\ \vdots \\ \hat{\mathbf{x}}_{\mathrm{K}}(t_{i+1}^-) \end{bmatrix} =
\begin{bmatrix}
(\mathbf{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c\,nom}}\sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}}\mathbf{K}_{\mathrm{j}}\mathbf{H}_{\mathrm{t}}) & -\mathbf{B}_{\mathrm{d\,t}}p_1\mathbf{G}^*_{\mathrm{c\,nom}}(\mathbf{I}-\mathbf{K}_1\mathbf{H}_1) & \cdots & -\mathbf{B}_{\mathrm{d\,t}}p_{\mathrm{K}}\mathbf{G}^*_{\mathrm{c\,nom}}(\mathbf{I}-\mathbf{K}_{\mathrm{K}}\mathbf{H}_{\mathrm{K}}) \\
(\mathbf{\Phi}_1\mathbf{K}_1\mathbf{H}_{\mathrm{t}} - \mathbf{B}_{\mathrm{d1}}\mathbf{G}^*_{\mathrm{c\,nom}}\sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}}\mathbf{K}_{\mathrm{j}}\mathbf{H}_{\mathrm{t}}) & \begin{bmatrix} \mathbf{\Phi}_1(\mathbf{I}-\mathbf{K}_1\mathbf{H}_1) \\ -\mathbf{B}_{\mathrm{d1}}p_1\mathbf{G}^*_{\mathrm{c\,nom}}(\mathbf{I}-\mathbf{K}_1\mathbf{H}_1) \end{bmatrix} & \cdots & -\mathbf{B}_{\mathrm{d1}}p_{\mathrm{K}}\mathbf{G}^*_{\mathrm{c\,nom}}(\mathbf{I}-\mathbf{K}_{\mathrm{K}}\mathbf{H}_{\mathrm{K}}) \\
\vdots & \vdots & \ddots & \vdots \\
(\mathbf{\Phi}_{\mathrm{K}}\mathbf{K}_{\mathrm{K}}\mathbf{H}_{\mathrm{t}} - \mathbf{B}_{\mathrm{d\,K}}\mathbf{G}^*_{\mathrm{c\,nom}}\sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}}\mathbf{K}_{\mathrm{j}}\mathbf{H}_{\mathrm{t}}) & -\mathbf{B}_{\mathrm{d\,K}}p_1\mathbf{G}^*_{\mathrm{c\,nom}}(\mathbf{I}-\mathbf{K}_1\mathbf{H}_1) & \cdots & \begin{bmatrix} \mathbf{\Phi}_{\mathrm{K}}(\mathbf{I}-\mathbf{K}_{\mathrm{K}}\mathbf{H}_{\mathrm{K}}) \\ -\mathbf{B}_{\mathrm{d\,K}}p_{\mathrm{K}}\mathbf{G}^*_{\mathrm{c\,nom}}(\mathbf{I}-\mathbf{K}_{\mathrm{K}}\mathbf{H}_{\mathrm{K}}) \end{bmatrix}
\end{bmatrix}
\begin{bmatrix} \mathbf{x}_{\mathrm{t}}(t_i) \\ \hat{\mathbf{x}}_1(t_i^-) \\ \vdots \\ \hat{\mathbf{x}}_{\mathrm{K}}(t_i^-) \end{bmatrix}
$$

$$
+\begin{bmatrix}
-\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c\,nom}}\sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}}\mathbf{K}_{\mathrm{j}} \\
(\mathbf{\Phi}_1\mathbf{K}_1 - \mathbf{B}_{\mathrm{d1}}\mathbf{G}^*_{\mathrm{c\,nom}}\sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}}\mathbf{K}_{\mathrm{j}}) \\
\vdots \\
(\mathbf{\Phi}_{\mathrm{K}}\mathbf{K}_{\mathrm{K}} - \mathbf{B}_{\mathrm{d\,K}}\mathbf{G}^*_{\mathrm{c\,nom}}\sum_{\mathrm{j=1}}^{\mathrm{K}} p_{\mathrm{j}}\mathbf{K}_{\mathrm{j}})_{\mathrm{j}}
\end{bmatrix}
\mathbf{v}_{\mathrm{t}}(t_i) +
\begin{bmatrix} \mathbf{G}_{\mathrm{d\,t}} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix}
\mathbf{w}_{\mathrm{d\,t}}(t_i) \qquad (5.35)
$$

As usual, this is can be expressed in the form:

$$
\begin{bmatrix} \mathbf{x}_{\mathrm{t}}(t_{i+1}) \\ \hat{\mathbf{x}}_1(t_{i+1}^-) \\ \vdots \\ \hat{\mathbf{x}}_{\mathrm{K}}(t_{i+1}^-) \end{bmatrix} = \mathbf{T}_{\text{nom-LB}}
\begin{bmatrix} \mathbf{x}_{\mathrm{t}}(t_i) \\ \hat{\mathbf{x}}_1(t_i^-) \\ \vdots \\ \hat{\mathbf{x}}_{\mathrm{K}}(t_i^-) \end{bmatrix} + \mathbf{L}_{\text{nom-LB}}
\begin{bmatrix} \mathbf{w}_{\mathrm{d\,t}}(t_i) \\ \mathbf{v}_{\mathrm{t}}(t_i) \end{bmatrix} \qquad (5.36)
$$

where $\mathbf{T}_{\text{nom-LB}}$ and $\mathbf{L}_{\text{nom-LB}}$ are expressed as:

$$\mathbf{T}_{\text{nom-LB}} = \begin{bmatrix} (\mathbf{\Phi}_{\text{t}} - \mathbf{B}_{\text{d t}}\mathbf{G}^{*}_{\text{c nom}}\sum_{\text{j=1}}^{\text{K}} p_{\text{j}}\mathbf{K}_{\text{j}}\mathbf{H}_{\text{t}}) & -\mathbf{B}_{\text{d t}}p_1\mathbf{G}^{*}_{\text{c nom}}(\mathbf{I}-\mathbf{K}_1\mathbf{H}_1) & \cdots & -\mathbf{B}_{\text{d t}}p_{\text{K}}\mathbf{G}^{*}_{\text{c nom}}(\mathbf{I}-\mathbf{K}_{\text{K}}\mathbf{H}_{\text{K}}) \\ (\mathbf{\Phi}_1\mathbf{K}_1\mathbf{H}_{\text{t}} - \mathbf{B}_{\text{d1}}\mathbf{G}^{*}_{\text{c nom}}\sum_{\text{j=1}}^{\text{K}} p_{\text{j}}\mathbf{K}_{\text{j}}\mathbf{H}_{\text{t}}) & \begin{bmatrix} \mathbf{\Phi}^1(\mathbf{I}-\mathbf{K}^1\mathbf{H}^1) \\ -\mathbf{B}_{\text{d1}}p_1\mathbf{G}^{*}_{\text{c nom}}(\mathbf{I}-\mathbf{K}_1\mathbf{H}_1) \end{bmatrix} & \cdots & -\mathbf{B}_{\text{d1}}p_{\text{K}}\mathbf{G}^{*}_{\text{c nom}}(\mathbf{I}-\mathbf{K}_{\text{K}}\mathbf{H}_{\text{K}}) \\ \vdots & \vdots & \ddots & \vdots \\ (\mathbf{\Phi}_{\text{K}}\mathbf{K}_{\text{K}}\mathbf{H}_{\text{t}} - \mathbf{B}_{\text{dK}}\mathbf{G}^{*}_{\text{c nom}}\sum_{\text{j=1}}^{\text{K}} p_{\text{j}}\mathbf{K}_{\text{j}}\mathbf{H}_{\text{t}}) & -\mathbf{B}_{\text{dK}}p_1\mathbf{G}^{*}_{\text{c nom}}(\mathbf{I}-\mathbf{K}_1\mathbf{H}_1) & \cdots & \begin{bmatrix} \mathbf{\Phi}_{\text{K}}(\mathbf{I}-\mathbf{K}_{\text{K}}\mathbf{H}_{\text{K}}) \\ -\mathbf{B}_{\text{dK}}p_{\text{K}}\mathbf{G}^{*}_{\text{c nom}}(\mathbf{I}-\mathbf{K}_{\text{K}}\mathbf{H}_{\text{K}}) \end{bmatrix} \end{bmatrix} \tag{5.37}$$

and

$$\mathbf{L}_{\text{nom-LB}} = \begin{bmatrix} \mathbf{G}_{\text{d t}} & -\mathbf{B}_{\text{d t}}\mathbf{G}^{*}_{\text{c nom}}\sum_{\text{j=1}}^{\text{K}} p_{\text{j}}\mathbf{K}_{\text{j}} \\ \mathbf{0} & (\mathbf{\Phi}_1\mathbf{K}_1 - \mathbf{B}_{\text{d1}}\mathbf{G}^{*}_{\text{c nom}}\sum_{\text{j=1}}^{\text{K}} p_{\text{j}}\mathbf{K}_{\text{j}}) \\ \vdots & \vdots \\ \mathbf{0} & (\mathbf{\Phi}_{\text{k}}\mathbf{K}_{\text{k}} - \mathbf{B}_{\text{dk}}\mathbf{G}^{*}_{\text{c nom}}\sum_{j=1}^{\text{K}} p_{\text{j}}\mathbf{K}_{\text{j}}) \end{bmatrix} \tag{5.38}$$

with the statistics for the noise as:

$$E\left\{\begin{bmatrix} \mathbf{w}_{\text{d t}}(t_i) \\ \mathbf{v}_{\text{t}}(t_i) \end{bmatrix}\begin{bmatrix} \mathbf{w}_{\text{d t}}{}^{\text{T}}(t_j) & \mathbf{v}_{\text{t}}{}^{\text{T}}(t_j) \end{bmatrix}\right\} = \begin{cases} \mathbf{Q}_{\mathbf{0}}(t_i) & t_i = t_j \\ \mathbf{0} & t_i \neq t_j \end{cases} \tag{5.39}$$

where

$$\mathbf{Q}_{\mathbf{0}}(t_i) = \begin{bmatrix} \mathbf{Q}_{\text{d t}}(t_i) & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{\text{t}}(t_i) \end{bmatrix}$$

The output autocorrelation of the augmented system at time t_{i+1} is written as:

$$\mathrm{E}\left\{\begin{bmatrix}\mathbf{x}^{\mathrm{t}}(t_{i+1})\\ \hat{\mathbf{x}}_1(t_{i+1}^-)\\ \vdots\\ \hat{\mathbf{x}}_{\mathrm{K}}(t_{i+1}^-)\end{bmatrix}\begin{bmatrix}\mathbf{x}_{\mathrm{t}}^{\mathrm{T}}(t_{i+1}) & \hat{\mathbf{x}}_1^{\mathrm{T}}(t_{i+1}^-) & \cdots & \hat{\mathbf{x}}_{\mathrm{K}}^{\mathrm{T}}(t_{i+1}^-)\end{bmatrix}\right\} = \boldsymbol{\Xi}_{\mathrm{k}}(t_{i+1}^-)$$

$$= \mathbf{T}_{\text{nom-LB}}\,\boldsymbol{\Xi}_{\mathrm{k}}(t_i^-)\,\mathbf{T}_{\text{nom-LB}}^{\mathrm{T}} + \mathbf{L}_{\text{nom-LB}}\mathbf{Q_0}\mathbf{L}_{\text{nom-LB}}^{\mathrm{T}} \tag{5.40}$$

Equation (5.40) is now used to give a performance assessment when there is a possible distribution of probabilities. Of course for the discretization of the parameter space, the cost function expressed in Equation (5.33) uses $\mathrm{E}\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\,\mathbf{y}\} \equiv \mathrm{tr}(\mathbf{W}\,\mathbf{C}\,\boldsymbol{\Psi}_{\mathrm{x_t}}\mathbf{C}^{\mathrm{T}})$ where $\boldsymbol{\Psi}_{\mathrm{x_t}} = \mathrm{E}\{[\mathbf{x}_{\mathrm{t}}][\mathbf{x}_{\mathrm{t}}]^{\mathrm{T}}\}$ from Equation (5.40). For evaluating the transient phase, $\mathrm{E}\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\,\mathbf{y}\}$ can be evaluated over the range of the possible probabilities. Of course, Equation (5.40) can be determined for a finite horizon, which is a better predictor of the transient phase. The point when steady state has been reached will determine that finite horizon.

5.3.1.4 Discussion

Implementation of MMAE-based control with a single controller gain is a special case of the MMAC and MMAE-based control with selected feedback control. It is similar to the latter but with only one selected control, and it is similar to the former under the condition that each elemental controller gain is identical. In fact, mathematically, the MMAC with a single nominal controller gain for each elemental filter is the same as the MMAE-based controller with a single controller gain.

The single advantage of the nominal controller MMAE-based control approach is the simplicity of its form. An algorithm to select the controller is not necessary and storing one controller gain matrix is minimal compared to other approaches for which there are multiple choices of controllers. However, the computer resource savings for implementation is not significant in comparison to other MMAE-based controllers. The number of operations to compute the control once the controller is specified will be the same for any other MMAE-based implementation.

Clearly, the disadvantage of the nominal controller approach is that the single gain must be robust enough to meet performance requirements for the entire range of the uncertain parameter space. One cannot hope to attain the same level of performance as is achievable by a full-scale MMAE-based controller. As pointed out before, the single controller gain is a reduced form of the MMAC. Thus, it offers less robustness than the MMAC.

The difference in computation and computer resource usage for the single controller gain and the MMAE-based control or MMAC control is not significant for most cases. Therefore, the single gain MMAE-based control may not the ideal approach for most control systems.

5.3.2 MMAE-Based Control with a Blended Controller Element

This section further develops the MMAE-based control with blended controller approach that was used as the basis of comparison with the MMAC in Section 5.2. The MMAE-based control from the previous section employed a single nominal controller and thus did not require a selection scheme for the controller. The blended controller approach uses several predesigned controllers but also does not use a selection scheme based on optimization of a cost function. As illustrated in Figure 5.4, the probability vector from the MMAE is used as a weighting on the controllers. The resulting gain matrix is then applied to the state estimate from the MMAE to obtain the control. Again, as was

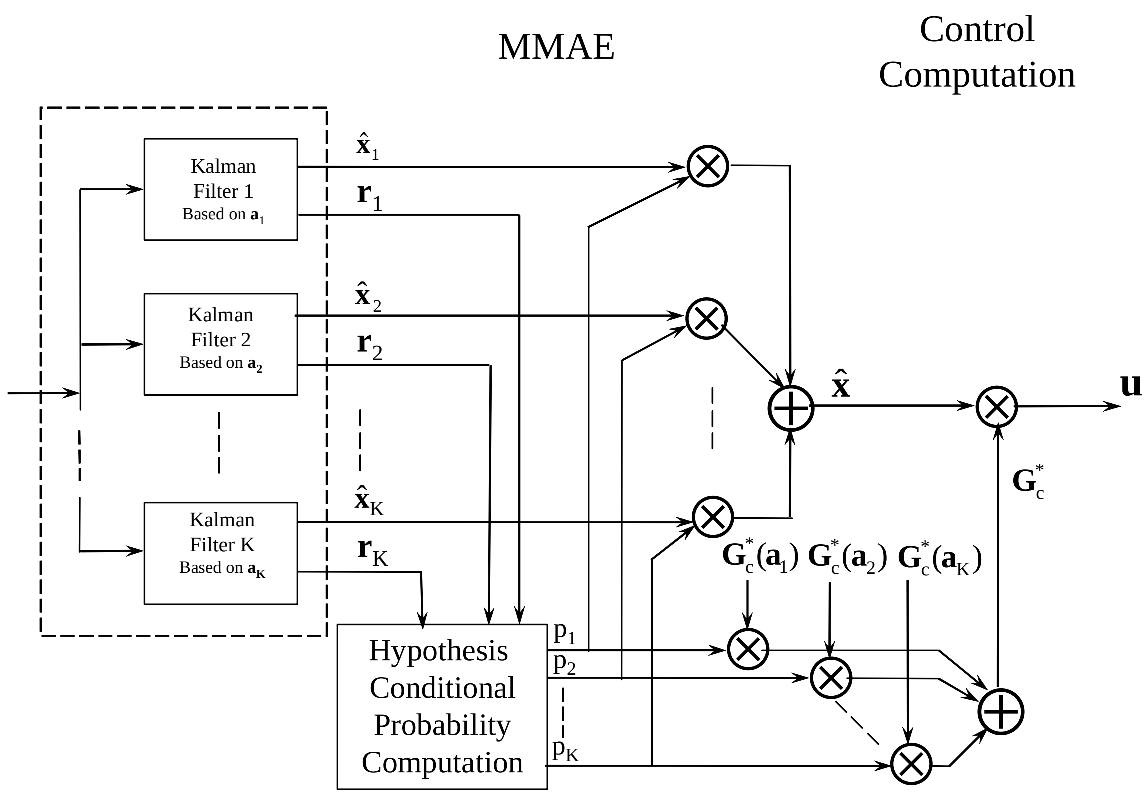
MMAE
Control Computation
Kalman Filter 1 Based on $\mathbf{a}_1$
Kalman Filter 2 Based on $\mathbf{a}_2$
Kalman Filter K Based on $\mathbf{a}_K$
$\hat{\mathbf{x}}_1$
$\mathbf{r}_1$
$\hat{\mathbf{x}}_2$
$\mathbf{r}_2$
$\hat{\mathbf{x}}_K$
$\mathbf{r}_K$
$\hat{\mathbf{x}}$
$\mathbf{u}$
$\mathbf{G}_c^*$
$\mathbf{G}_c^*(\mathbf{a}_1)$ $\mathbf{G}_c^*(\mathbf{a}_2)$ $\mathbf{G}_c^*(\mathbf{a}_K)$
Hypothesis Conditional Probability Computation
p_1
p_2
p_K

Figure 5.4 MMAE-based control with blended gain matrices

reviewed, this is similar to the MMAC, except in that case, the gain matrices corresponding to the filters are applied to the individual state estimates and then the probability weights are applied to yield the control. It follows that the design for the MMAE-based control with blended control will be very similar to the MMAC.

5.3.2.1 Performance Evaluation Equations

The development in this section follows the approach in the previous section in deriving the performance equations. This is an implementation of the blended control element used in the MMAE-based control comparison to the MMAC from Section 5.2. The expression for the control is a probability weighting applied to the possible full-state feedback controller gain matrices that are associated with the filters in the MMAE. The controller equation is expressed as:

$$\mathbf{G}^*_{\text{c-blend}} = \sum_{\text{k}=1}^{\text{N}} p_{\text{k}} \mathbf{G}^*_{\text{c}}(\mathbf{a}_{\text{ck}}) \tag{5.41}$$

The resultant control also developed in the MMAC comparison section is expressed as:

$$\begin{aligned}\mathbf{u} = -\mathbf{G}^*_{\text{c blend}}(p_1\hat{\mathbf{x}}_1(t_i^+) + p_2\hat{\mathbf{x}}_2(t_i^+)\cdots p_{\text{N}}\hat{\mathbf{x}}_{\text{N}}(t_i^+))\\ -[p_1\mathbf{G}^*_{\text{c}}(\mathbf{a}_{\text{c}1}) + p_2\mathbf{G}^*_{\text{c}}(\mathbf{a}_{\text{c}2})\cdots p_{\text{N}}\mathbf{G}^*_{\text{c}}(\mathbf{a}_{\text{cN}})](p_1\hat{\mathbf{x}}_1(t_i^+) + p_2\hat{\mathbf{x}}_2(t_i^+)\cdots p_{\text{N}}\hat{\mathbf{x}}_{\text{N}}(t_i^+))\end{aligned} \tag{5.42}$$

Note that, in Equation (5.41), the predesigned controllers $\mathbf{G}^*_{\text{c}}(*)$ are fixed. Thus, the control is actually a function of the probability. As has been assumed previously, one of the filters will have a probability of one at steady state (if lower bounds are not imposed on the computed probabilities). In this scheme, the filter probability vector is used not only as the weight for the state estimate in the MMAE, but for the controller gain as well. Thus, if say the kth filter has assumed all the probability at steady state, the control is expressed as:

$$\mathbf{u}(t_i) = -\mathbf{G}^*_{\text{c}}(\mathbf{a}_{\text{ck}})\,\hat{\mathbf{x}}_{\text{k}}(t_i^+) \tag{5.43}$$

Under these conditions, the control is identical the MMAC. Hence, as was the case for the nominal control, the performance evaluation will be the same as the MMAC. Thus the state equations are given by Equation (5.27) with $\mathbf{G}^*_{\text{c}}(\mathbf{a}_{\text{ck}})$ substituted for $\mathbf{G}^*_{\text{c}_{\text{nom}}}$, and the output correlation is given by Equation (5.32). With the controller substitution $\mathbf{G}^*_{\text{c}}(\mathbf{a}_{\text{ck}})$, Equation (5.32) can now be used in the expression of the cost function that will be evaluated during the discretization. The expected value of the autocorrelation of the states for the true system is $\mathbf{\Psi}_{\mathbf{x}_{\text{t}}\,\text{blend}} = \text{E}\{[\mathbf{x}_{\text{t}}][\mathbf{x}_{\text{t}}]^{\text{T}}\}$.

5.3.2.2 Discretization Algorithm

The fact that the steady state autocorrelation for the MMAE-based control with a blended controller element is identical to the MMAC is an indication that the discretization method will be identical. For the previous nominal controller case, the form of the performance measure was the same but the controller was not linked to the filter. The important aspect for the blended controller approach is that the controller is designed for the filter location, as is the case for the MMAC. Thus, in the discretization process, the controller is known and is *linked* to the filter.

Discretization is a matter of determining the optimal locations in parameter space for the possible filter/controller combinations that will minimize the cost based on the performance measure. The cost function minimized is the output squared over the range of possible values of the parameters and is expressed as:

$$J_{2\ \text{c-blend}} \equiv \frac{\int_A E\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y}\}da}{\int_A da} \tag{5.44}$$

where $E\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\,\mathbf{y}\} \equiv \mathrm{tr}(\mathbf{W}\,\mathbf{C}\;\boldsymbol{\Psi}_{\mathbf{x}_t\ \text{blend}}\mathbf{C}^{\mathrm{T}})$ and $\boldsymbol{\Psi}_{\mathbf{x}_t\ \text{blend}} = E\{[\mathbf{x}_t][\mathbf{x}_t]^{\mathrm{T}}\}$. The cost expressed in Equation (5.44) is, of course, identical to that used for the MMAC. Also, the discretization algorithm is the same as the MMAC and the steps are repeated here as follows:

1) Describe in terms of the parameter vector **a**, the truth model of the system, and the filter/controller.

2) Choose the number of filters K in the MMAE.

3) Choose a representative parameter set, $\mathcal{A} \equiv \{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_K\}$ to begin the minimization.

4) Use a numerical integration technique to evaluate or $J_{2\text{ c-blend}}$. This evaluation depends on determining to which filter the MMAE-based control will converge. Determination of the convergence is discussed in Section 2.5.3.

5) Use a vector minimization technique with the functional evaluation from Step (4) to minimize $J_{2\text{ c-blend}}$.

5.3.2.3 Probability Lower Bounding and Transient Response Evaluation

Bounding the lower possible probability on the filters addresses two different issues for this MMAE-based control approach: the design for preventing probability lockout and an assessment of the transient response. By deriving a performance measure assuming that there is a lower bound on the probability assigned for each filter in the MMAE, the optimal discretization can be performed with the same algorithm from the previous section. As discussed previously in Chapter 4, implementing lower bounding on the filter probabilities prevents filter lock-out at steady state. As was the case in the previous applications, to implement this lower bound, assume all the filters will have a probability of p_{min}, except for one filter. That filter will have a probability expressed as $p_{sel} = 1 - p_{min}(K-1)$. However, for this research, the lower bounding also provides the ability to assess the performance of the architecture in the transient phase.

From the previous discussion, it appears that the MMAC and the MMAE-based control with a blended controller element will have the same steady-state performance or, in other words, performance when one of the filters has assumed all the probability (assuming no lower bound on the computed probabilities). Since, mathematically the two approaches are different, as demonstrated in Section 5.2, the differences are apparent when more than one filter has non-zero probability. To assess the performance of the architecture with various possible probability values across the filters, the output autocorrelation equations with lower bounding from the MMAC are used.

The derivation of the autocorrelation equations for implementing lower bounding again borrows the results from the MMAC development. From Equation (5.42), it is clear that the control $\mathbf{G}^*_{c_{\text{blend}}}$ is applied to each state estimate, whereas for the MMAC the control gain matrix associated with the individual filter is used. Thus, the only modification to the MMAC approach is that the same control, $\mathbf{G}^*_{c_{\text{blend}}}$, is applied to the state vectors. The state equations with lower bounding are given by Equation (5.35) with $\mathbf{G}^*_{c_{\text{blend}}}$ substituted for $\mathbf{G}^*_{c_{\text{nom}}}$.

Equation (5.40) with $\mathbf{G}^*_{c_{\text{blend}}}$ substituted for $\mathbf{G}^*_{c_{\text{nom}}}$ is used to give a performance assessment when there is a possible distribution of probabilities. For any given value of the probability vector, there is an associated parameter estimate of the system. For the evaluation of Equation (5.40), it will be assumed that the parameter estimate reflects the current system and will be used in the true system model equations. Finally, a finite horizon will be used in the evaluation. Since the goal is to determine performance in the transient phase, solving the Lyapunov equation to steady state would not be an accurate assessment of the possible transient conditions. It will be up to the designer to determine when steady state is essentially reached and then choose over what length of period the transient response assessment is to occur.

Equation (5.40) evaluates the performance of the MMAE-based control with blended controller and can be used in the discretization algorithm from the previous section, if an assumed lower bounding is to be implemented. However, using Equation

(5.40) for improvement of the transient response is not possible for this controller architecture. This is because the controller gain matrices are fixed designs and the filter discretization is set during the initial design process for the steady state response. There are no other variables to change in architecture. The MMAE-based control architecture in the following sections addresses improvements to the transient response.

5.3.2.4 Discussion

Clearly, the MMAE-based control with a blended controller element is the next level of refinement over the nominal controller approach in the previous sub-section. However, this architecture does not offer much beyond the typical MMAC. The control still is determined by a blending of the control gains by a probability-based weighting scheme. The number of possible probability weights corresponds to the number of controller gains, which of course in turn is the number of filters. The only difference between this approach and the MMAC is that, for this approach, the state estimates are formed and then multiplied by a blended controller gain rather than multiplying the state estimates by the controller gains and then blended the results (blended LQG control). This difference was discussed fully in Section 5.2.

As with the MMAC, it is not possible to predict the transient state performance. However, by using the lower bounding equations, it is possible to compute an envelop of performance. This technique can also be applied to the MMAC to determine if the bounds on transient performance are comparable to those of similar MMAE-based control architectures. As was also discussed in Section 5.2, the steady state performance will be the same as that of the MMAC.

Mathematically, the MMAE-based control with a blended gain has fewer matrix multiplies than the MMAC. For larger scale problems, this difference may be significant. However, if it is a matter of performance, unless either the MMAE-based control with a blended control element or the MMAC yields a better transient performance, there is really not a significant reason to choose one architecture over the other.

5.3.3 MMAE-Based Control with Selected Controller Element

In this approach, the MMAE-derived state estimation feeds a full-state feedback controller gain that is *designed on the fly*, by invoking the separation principle of adaptive control [36]. Rather than performing the design of a full-state feedback controller gain, a table look-up scheme is developed. As previously discussed, there are two approaches to implementing a controller selection scheme: the first uses the parameter estimate, and the second incorporates the probability vector. State and parameter estimation from an MMAE are established approaches, as reviewed in Chapter 2. Both the parameter estimate and the probability vector that is used to form the estimates are available from the MMAE. Figure 5.5 illustrates the MMAE-based control architecture that uses the probability vector inherent in the estimation computations for the controller gain selection. Revisiting the previous section, the implementation in Figure 5.4 is a special case of the implementation in Figure 5.5. In both representations, the probability vector is used to determine the control gain matrix. Figure 5.5 is a generalization that conveys that the determination of the filter may use an algorithm-based selection scheme rather than multiplication of probabilities and gains. Use of the probability vector to *select* the control is a novel approach proposed herein.

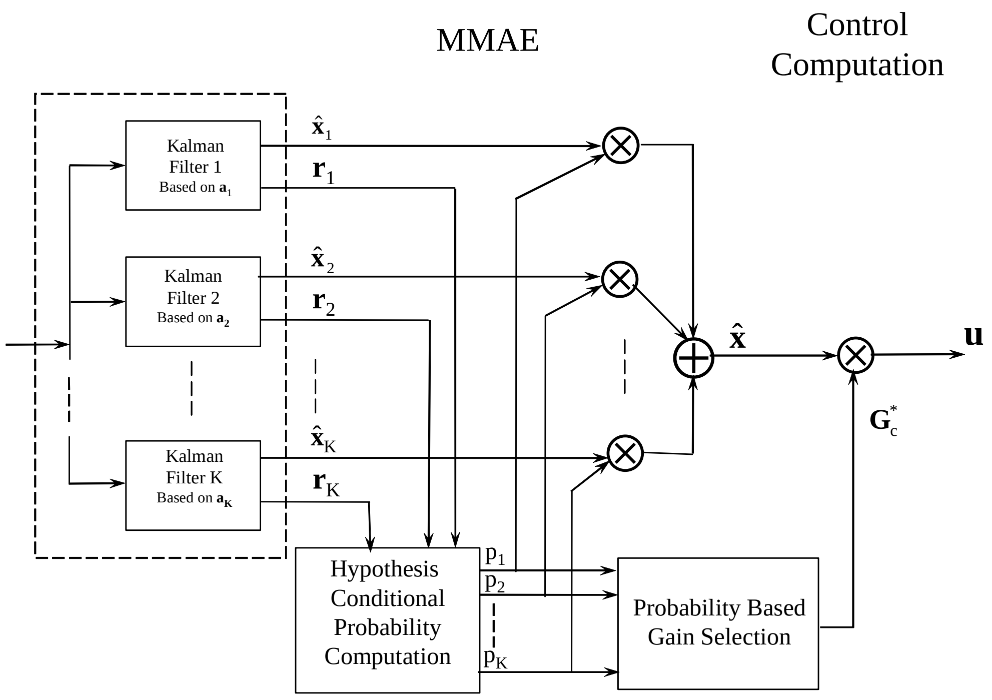
MMAE
Control Computation
Kalman Filter 1 Based on $\mathbf{a}_1$
Kalman Filter 2 Based on $\mathbf{a}_2$
Kalman Filter K Based on $\mathbf{a}_K$
$\hat{\mathbf{x}}_1$
$\mathbf{r}_1$
$\hat{\mathbf{x}}_2$
$\mathbf{r}_2$
$\hat{\mathbf{x}}_K$
$\mathbf{r}_K$
Hypothesis Conditional Probability Computation
p_1
p_2
p_K
Probability Based Gain Selection
$\hat{\mathbf{x}}$
$\mathbf{G}_c^*$
$\mathbf{u}$

Figure 5.5 MMAE-based control with probability based controller selection

In the previous design approaches, it was assumed the performance measure corresponded to steady state conditions. Also the controllers are fixed and do not adjust depending on the state of the system. Change in the applied control is regulated by external factors such as using the probability as a weight. For a table look-up approach, there is more flexibility for the control to be designed for specific assumed operating points. This flexibility of the table look-up approach has the potential to improve performance.

5.3.3.1 Performance Evaluation Equations

As was the case for the MMAC and MMAE-based controller with a nominal control element, output autocorrelation performance is evaluated at steady state. As is typical, one filter will assume all the probability at steady state if lower bounds are not imposed on the computed probabilities. Thus the control at steady state is expressed as:

$$\mathbf{u}(t_i) = -\mathbf{G}_{\mathrm{c}}^{*}(t_i, sel)\hat{\mathbf{x}}_{\mathrm{MMAE}}(t_i^{+}) \tag{5.45}$$

The control gain is defined by a table look-up scheme driven by one of the two proposed methods. As shown in Equation (5.45), the controller gain is denoted as a function of *sel*; *sel* is (an abuse of) notation to indicate the algorithm that is used to select the controller gain. As will be discussed, there is more that one potential selection algorithm to investigate and so *sel* is meant to be generic. The actual selection scheme is not important at this point, but just the fact that a single controller is selected. Now denote the gain $\mathbf{G}_{\mathrm{c}}^{*}(t_i,sel)$ as $\mathbf{G}_{\mathrm{c}sel}^{*}$. The form of the control expressed in Equation (5.45) is the same as developed for the nominal controller. It follows that the form of the steady state autocorrelation will be the same. Thus, the state equations are given by Equation (5.27) with $\mathbf{G}_{\mathrm{c}sel}^{*}$ substituted for $\mathbf{G}_{\mathrm{cnom}}^{*}$ and the output correlation is given by Equation (5.32). With the controller substitution, Equation (5.32) can now be used in the expression of the cost function that will be evaluated during the discretization. The expected value of the autocorrelation of the states for the true system is $\boldsymbol{\Psi}_{\mathbf{x}_t} = \mathrm{E}\{[\mathbf{x}_t][\mathbf{x}_t]^{\mathrm{T}}\}$.

Given that the controller gain in Equation (5.48) can be selected on the fly, the question becomes how to design the feedback controller gain matrix a priori. For a fixed uncertain parameter space, i.e., one that is not set up as a moving bank, there are a fixed number of controllers based on a discretization of that space. However, as has been found in previous work [56,57], the best discretization for control is identical to discretization of the uncertain space for state estimates. However, the MMAE will not provide the best parameter estimate and the best state estimate simultaneously. For the table look-up selection approach, the best parameter estimate is not required. As long as the parameter estimate from the MMAE is consistent, then the parameter estimate can be a pointer into the table and the controller at the table entry actually may not be based on the parameter estimate, but a gain matrix that corresponds to the state estimate.

Since the steady state result for the MMAE-based control is the same as the MMAC, the discretization for the filters is essentially the same as the MMAC implementation. But the versatility of the MMAE-based control allows an arbitrarily finite number of controller gains to be in the controller gain selection table. The size of the look-up table is constrained only by physical implementation restrictions. The points in the table that correspond to the filter locations will obviously be the controllers used in the evaluation of the steady state performance used for the discretization. Since the table contains more points than the number of filters in the MMAE, other entries in the table need to be determined for when the MMAE has not reached steady state, or in other words, for the transient response.

5.3.3.2 Probability Lower Bounding and Transient Response

Like the MMAE-based control structure that employs probability-based blending for the control element, probability lower bounding can be used to prevent filter lock-out, and the performance equations can be used to evaluate the transient response. However, unlike the previous implementation of MMAE-based control, selecting the controller has the potential of improving the transient response, since the control gains are not limited are not fixed corresponding to filter parameter locations. As in the previous case, before the MMAE reaches steady state, there will be a distribution of probabilities across the elemental filters. Of course, those probabilities are used to determine the parameter and state estimates for the current operating point. The parameter estimate can be used in the controller selection or the probability vector can be used as an index into a gain look-up table. In either case, the performance can be assessed during the transient response when there is a distribution of probabilities.

The derivation of the autocorrelation equations for implementing lower bounding follows the previous derivation and again borrows the results from the MMAC development. Thus, rather than $\mathbf{G}_{\mathrm{c}}^{*}$ applied to each state estimate as in Equation (5.42), the performance evaluation uses $\mathbf{G}_{\mathrm{c}sel}^{*}$ as in Equation (5.45). The state equations with lower bounding are given by Equation (5.35) with $\mathbf{G}_{\mathrm{c}sel}^{*}$ substituted for $\mathbf{G}_{\mathrm{cnom}}^{*}$. Equation (5.40) with $\mathbf{G}_{\mathrm{c}sel}^{*}$ substituted for $\mathbf{G}_{\mathrm{cnom}}^{*}$ is used to give a performance assessment when there is a possible distribution of probabilities.

5.3.3.3 Discretization Algorithms

The discretization algorithm is the point in the development where the selection scheme for the controller gain implementation becomes important. The performance evaluation for both steady-state and the transient response are independent of the controller selection scheme. For the MMAE-based control with a selected controller element, there are actually two separate discretizations. The first discretization is for the MMAE portion, as is typically done. The second discretization populates the look-up controller gain table. It is the definition of this table and the indexing methods into the table that is dependent on the selection scheme.

The discretization for the MMAE is also independent of the table look-up scheme implementation. Of course it is assumed that the controller used in the discretization provides the best performance using the same criterion used for the rest of the controllers in the gain look-up table. Once the parameter space has been discretized, the entries in the table corresponding to the filter information are incorporated into the table. The mapping of the decision information (such as parameter estimates) to the gain in the table is the selection scheme.

As with the two previous MMAE-based control approaches, the MMAE portion is discretized using the same approach as was used for the MMAC. The objective is still the same: discretize the parameter space such that the steady-state performance is optimized. Again, this is accomplished by minimizing the cost function:

$$J_{2\ \text{c-ss}} \equiv \frac{\int_A E\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y}\}da}{\int_A da} \tag{5.46}$$

where $E\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\,\mathbf{y}\} \equiv \mathrm{tr}(\mathbf{W}\,\mathbf{C}\,\mathbf{\Psi}_{\mathbf{x}_t}\mathbf{C}^{\mathrm{T}})$ and $\mathbf{\Psi}_{\mathbf{x}_t} = E\{[\mathbf{x}_t][\mathbf{x}_t]^{\mathrm{T}}\}$. The algorithm to discretize the parameter space is similar to the Modified MMAC and is given as:

1) Describe in terms of the parameter vector $\mathbf{a}$, the truth model of the system, and the filter. Describe the corresponding full-state feedback controller gain in terms of $\mathbf{a}_c$.
2) Choose the number of filters K in the MMAE.
3) Choose a representative parameter set, $\mathcal{A} \equiv \{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_K\}$ to begin the minimization and design a full state feedback controller gain corresponding to the filter, but based on the design point $\mathbf{a}_c$.
4) Use a numerical integration technique to evaluate Equation (5.46).
    a. Compute $\mathbf{\Xi}_{k\infty}$ using Equation (5.32) at discrete points in the parameter space (for $\mathbf{a}_t$ value) for each filter: k = 1,…K.

b. At each discrete point, evaluate the proximity measure using $\mathbf{\Xi}_{k\infty}$ for k = 1,…K to determine the convergence to a single elemental filter/controller with the maximum probability at steady state. Denote that selected filter/controller as *sel*.

c. For the selected filter/controller, determine $\mathbf{\Psi}_{\mathbf{x}_t\, sel}$ from the upper right partition of $\mathbf{\Xi}_{sel\infty}$ saved from the previous evaluation of $\mathbf{\Xi}_{k\infty}$ for k = 1,…K and use in the evaluation of the cost function.

5) Use a vector minimization technique with the functional evaluation from Step (4) to minimize $J_{2\ c\text{-}ss}$.

It is assumed that the controllers for this implementation algorithm yield the best control at steady state for each filter location in parameter space. The best control is obtained by using the techniques discussed in Chapter 3. The controller design is not necessarily dependent on the parameters used to define the filter to which the controller corresponds. Now, of course, this result has to be incorporated into the table look-up scheme implementation. After the steady-state discretization, only the entries mapped to the MMAE filters are defined in the gain look-up table. The remaining entries, which correspond to the transient phase, are determined by an implementation specific to the table look-up approach. The following sub-sections discuss the discretization for the probability-based and parameter estimation approaches to table look-up.

5.3.3.4 Controller Selection by Probability-Based Table Look-up

Controller selection using a probability based table look-up approach maps the current state of the probability vector to the index in the table of full-state feedback gains. To implement this selection scheme, the table indexing method must be defined and the population of the table must be accomplished. The previous section determined the discretization of the MMAE portion of the architecture, and those corresponding controller gains are used in the table. With those entries determined, the final step to complete the gain table is to populate all remaining entries for all points not considered steady state.

The probability based look-up table is organized according to the number of filters and the number of elements per dimension of the table. The number of filters determines the dimension of the table. For example, an MMAE-based controller with three filters will have a three-dimensional look-up table. The number of elements in each dimension will determine the *resolution of the probability*, which is 1/(number of elements –1). Thus, continuing the example, if there are 11 elements for each index, the table size will have 11^3 entries and each index corresponds to an increment in probability of 1/10 (e.g. 0, 1/10, 2/10…1). Though the number of dimensions of the table is set according to the size of the probability vector, its size is limited only by physical design constraints on the implementation.

The first step to populate the look-up table is to map the steady-state discretization points to the entries in the look-up table. The maximum index value corresponds to the maximum available probability. Continuing the previous example, assume that at steady state the possible probabilities are: {(1,0,0) (0,1,0) (0,0,1)}. Assuming that there are eleven elements for each dimension and they are arbitrarily numbered 0…10, the corresponding table indices will be (10,0,0), (0,10, 0), and (0,0,10) pointing to controller gains $\mathbf{G}_c^*(a_1)$, $\mathbf{G}_c^*(a_2)$, and $\mathbf{G}_c^*(a_3)$, respectively. The filter locations

a_1, a_2, and a_3 are determined by discretization and $\mathbf{G}_c^*(*)$ is the controller gain designed for those locations.

Since the filter locations are set, the look-up table controller gains corresponding to the probabilities, (p_1, p_2, ... p_N) are determined by finding the gain that minimizes the performance measure based on the state equations for the lower bounding of the probabilities. In this case the cost function is simply:

$$J_{2_{\text{C-Transient}}} = E\{\mathbf{y}^T \mathbf{W}\, \mathbf{y}\} \equiv \operatorname{tr}\left(\mathbf{W}\, \mathbf{C}\, \mathbf{\Psi}_{\mathbf{x}_t} \mathbf{C}^T\right) \tag{5.47}$$

where $\mathbf{\Psi}_{\mathbf{x}_t} = E\{[\mathbf{x}_t][\mathbf{x}_t]^T\}$ is the autocorrelation evaluated using Equation (5.50). The subscript for the cost function expressed in Equation (5.47) denotes the period of the transient response, during which it is assumed that the filters have an associated nonzero probability (and not equivalent to an arbitrarily defined lower bound). Recall that, at steady state, all filters except for one will have zero probability (or an arbitrary set lower bound).

Since the cost function in Equation (5.47) is an evaluation of the transient response, a finite horizon should be considered for determining the state autocorrelation. Solving the Lyapunov equation determines the steady state response. However, the state autocorrelation can be evaluated over a finite period to determine an average performance over that period. This will give a better assessment of the performance up to a point that is to be considered steady state, rather than just evaluating the steady state response (with the assumption of nonzero probabilities on the filters). Again, except for setting an arbitrary lower bounding, nonzero probabilities on the filters will not occur at steady state, except for the one that has absorbed all the probability. Also contributing to the decision to use a shorter period over which to evaluate the cost is the fact that the probability weights across the filter will change values in the process of reaching steady state. Thus, it will be up to the designer to determine the finite horizon for evaluation, of which a major factor is the point that is to be considered steady state.

Now, to fill in the gain table, it is simple a matter of determining the gains that optimize Equation (5.47) corresponding to the indices for the table that has been set up according to its dimensions. This can be summarized with the following algorithm:

1) Describe in terms of the probability vector and controller parameter $\mathbf{a}_c$, the truth model of the system and the filter/controller to be used in Equation (5.40).
2) Determine the number of control look-up table entries from the size of the probability vector and the number of elements for each index.
3) Determine the controller that minimizes the cost function $J_{2\ \text{C-Transient}}$. Apply the *search across controller models* technique described in Chapter 3 or any other applicable approach.
4) Repeat step 3 for each valid index into the control table.

This four-step algorithm completes the look-up gain table for the design of the MMAE-based control using the probability vector as the selection method.

Note that step four of the population algorithm implies that only valid entries need to be considered. The fact is that a significant number of table entries will *not* be valid. Consider the previous illustrative example that has a table with 11^3 entries. The steady state discretization only fills three of them and the population algorithm must fill the rest. Recall that the probabilities must add to one. For example, (7/10, 2/10, 1/10) corresponding to the index (7,2,1) is valid, whereas the probability vector (1/10, 1/10, 1/10) is not possible and the table entry (1,1,1) need not be considered.

5.3.3.5 Controller Selection by Parameter Estimate Table Look-up

Whereas the previous section uses the probability vector as a means to index into a table of full-state feedback control gains, the approach outlined in this section simply uses the parameter estimate. This selection scheme maps the parameter estimate to the table index. Now, though there is only one mapping from the parameter estimate to the table, as will be described, the determination of the full state feedback gain in the table can be done in two different ways. One approach uses the parameter estimate as the assumed design parameter, and the second approach incorporates the possible probability vectors that form the parameter estimate, so it uses the computed $p_{\mathrm{k}}(t_i)$ values rather than the resulting $\hat{\mathbf{a}}\,(t_i)$ to perform the design.

The map from the parameter estimate to the index in a single dimension is a scaling operation and is expressed as:

$$\text{index} = \text{Round}\left(\frac{\text{Parameter Value - Parameter Min}}{\text{Parameter Range}}\right) * \text{Size of Index} \qquad (5.48)$$

Equation (5.48) is applied for each parameter that defines the filter/controller models. The number of dimensions of the table is equal to the number of different parameters. It is assumed that the parameter values placed during the initial discretization are in the defined parameter value range and will correspond to entries in the table. Of course, this only takes care of the number of entries in the look-up table corresponding to the number of filters. The remaining entries must map the parameter estimate to a controller.

Actually, to populate the table of controller gains, it is best to determine the map from the index to the parameter estimate. This function is, of course, the inverse of Equation (5.48) and is expressed as:

$$\text{Parameter Value} = \left(\frac{\text{index}}{\text{Size of Index}}\right)\text{Parameter Range - Parameter Min} \tag{5.49}$$

Equation (5.49) is used for each dimension or in other words, each of the different parameters that describes the filter/controller models. The task of populating the gain table now becomes a matter of determining the parameter value for each element of the index and performing a design for those values.

The first method to design the full-state feedback controller gain simply uses the parameter value as the design parameter and specifies a controller that yields the best control. The parameter value is, of course, the parameter estimate from the MMAE portion of the architecture. It is assumed that this parameter estimate reflects the true system and is used in the performance evaluation. The second major assumption is the parameter value for the filter. Since steady state has not been reached, a single filter of the MMAE filters does not form the state estimate, rather there is a blending of the state estimates from all the filters. Therefore, it will be assumed that the filter model to be used in the performance evaluation is derived from the parameter estimate. It is acknowledged that the discretization for the MMAE was for the best control and not the best parameter estimate. With these assumptions, the following algorithm is proposed to populate the gain table:

1) Describe in terms of the parameter vector $\mathbf{a}$ (derived from the index value), the truth model of the system and the filter and the controller in terms of the parameter $\mathbf{a}_c$.
2) Choose a representative parameter $\mathbf{a}_c$ to begin the minimization.
3) Use LQG design techniques to design controller to minimize $J_{2c\text{-}ss}$
4) Repeat 3 for each table entry.

Populating the table using the parameter estimate as the assumed true parameter and the basis for the filter model is not an exact representation of what may physically occur. In fact, the parameter estimate may be formed by one of a number of combinations of the probabilities and parameter values of the MMAE filters.

The second method of determining the table entries compensates for the fact that, for a single $\hat{\mathbf{a}}$, there corresponds a set of possible probability vectors that could have yielded that $\hat{\mathbf{a}}$. Now, since the probability vector also forms the state estimate, then there also could be entire set of state estimates $\hat{\mathbf{x}}$, that correspond to that single $\hat{\mathbf{a}}$. The control that is determined by the parameter estimate may be not the best for the actual instance of the state estimate. Thus, for each possible table entry that is indexed by the parameter estimate, a design procedure that averages the performance over the possible probability vectors is proposed.

To determine the best control for each parameter estimate $\hat{\mathbf{a}}$ in the parameter space, the following cost function is proposed:

$$J(\hat{\mathbf{a}}) \equiv \frac{\int_{\mathcal{P}} E\{\mathbf{y}^T \mathbf{W} \mathbf{y}\} dp}{\int_{\mathcal{P}} dp} \tag{5.50}$$

where $E\{\mathbf{y}^T \mathbf{W} \mathbf{y}\} \equiv \mathrm{tr}(\mathbf{W} \mathbf{C} \boldsymbol{\Psi}_{\mathbf{x}_t} \mathbf{C}^T)$ and $\boldsymbol{\Psi}_{\mathbf{x}_t} = E\{[\mathbf{x}_t][\mathbf{x}_t]^T\}$ is the autocorrelation evaluated using lower bounding of the probability expressed in Equation (5.40). The space $\mathcal{P}$, over which the cost is evaluated is all the possible probability vectors such that $\hat{\mathbf{a}} = \boldsymbol{p}\mathbf{I}\mathbf{a}$ where $\mathbf{a}$ has been determined previously by the discretization of the MMAE portion of the architecture. Assuming that discrete values of the probability vector will be used, Equation (5.50) can be evaluated as:

$$J(\hat{\mathbf{a}}) \equiv \frac{1}{N}\sum_{i=1}^{N} E\{\mathbf{y}^{T}\mathbf{W}\mathbf{y}\}\Big|_{\boldsymbol{p}_i} \forall \boldsymbol{p}_i \in \mathcal{P} \mid \hat{\mathbf{a}} = \boldsymbol{p}_i\mathbf{a} \tag{5.51}$$

where N is the number of probability vectors from the space of possible probably vectors $\mathcal{P}$, that compute the specified $\hat{\mathbf{a}}$.

For each possible parameter estimate that will be indexed into the controller gain table, the gain matrix that minimizes the cost function expressed in Equation (5.51) will be determined. This is accomplished by the following algorithm:

1) Describe in terms of the probability vector $\boldsymbol{p}$ and the parameter $\mathbf{a}$, the truth model of the system and the filter, and the parameter $\mathbf{a}_c$ for the controller.
2) Choose a representative parameter $\mathbf{a}_c$ to begin the minimization.
3) Determine the possible probability vectors for the assume parameter estimate and evaluate $J(\hat{\mathbf{a}})$.
4) Use a vector minimization technique with the functional evaluation from Step (3) to minimize $J(\hat{\mathbf{a}})$.
5) Repeat steps (1)-(4) for each possible parameter estimate that will index the controller look-up gain table.

The design of the controller gain using the above algorithm is very similar to the enhanced robustness approach. Given only the parameter estimate, it cannot be ascertained which probabilities form the estimate. The numerous possible probability weightings also mean that there are various possible state estimates, which may require different control. Thus, using the proposed approach, performance is essentially averaged over the possible probability vector values.

5.3.3.6 Discussion

The MMAE-based control with selected controller element is the next level of refinement over the previous two architectures. In fact, both the nominal controller element and the blended controller element are special cases of the selected controller element approach. Clearly, if all entries in the look-up table that is used for the selection scheme contain the same (nominal) controller element, then that would be equivalent to the nominal controller architecture. Now, if the look-up table is sized and indexed based on the probability vector and the gains are derived from blending gains according to the probabilities, then that would be equivalent to the MMAE-based control with blended control. As is typical with most control architecture designs, the general case architecture should provide more versatility and performance at least equivalent to, if not better than, the special cases.

Of the two approaches for selection schemes presented, using the parameter estimate as an index is a special case of using the probability vector. This is a corollary to the fact that the parameter estimate is derived from the probability vector. Thus, using the probability vector as the selection scheme is the general case and, as such, is more versatile and performance is at least equivalent to, and if not better than, using the parameter estimate as the selection scheme.

The disadvantage of using the probability vector as the basis of the selection scheme is that the size of the table most likely will be larger than for the parameter estimate approach. Of course, the relative size of the implementation of the two tables depends on how fine the increments for the parameter estimate are and the probabilities. The size of the table is a physical constraint that must be considered for implementation.

If table size is not an implementation constraint, then the more general case of the probability look-up table would be the best choice simply because of its versatility.

Finally, since it was shown that the MMAE-based control is the same as MMAC at steady-state, it is only the performance during the transient that may show any improvements of one form versus the other. It is not possible to get a prediction of the performance other than at steady-state because of the nature of the stochastic control. Thus, in order to assess if there is any performance improvement in the transient phase, a sufficient number of Monte Carlo simulations will be have to be run. It will be up to the designer to determine the transient period and over which the performance of the Monte Carlo simulations will be assessed.

5.3.4 MMAE-Based Control with an LQG Controller Element

As was pointed out in Chapter 2, an MMAE cannot be designed to provide the best state estimate and the best parameter estimate simultaneously. MMAE-based control with a selected LQG controller element as shown in Figure 5.6 partially addresses this issue. For all the previously discussed MMAE-based control architectures, the state estimates are derived from blending the individual estimates from the elemental filters within the MMAE structure. The residuals from individual filters are used to derive the probability weighting for the blending of the state estimate and the controller selection. For those architectures, the intent is not to provide the best parameter estimate, but to provide the best control.

The intent of the MMAE-based control with an LQG controller element is to provide the best parameter estimate that corresponds to the best control. The fact that the optimal discretizations for the best parameter estimation and the best control are not the

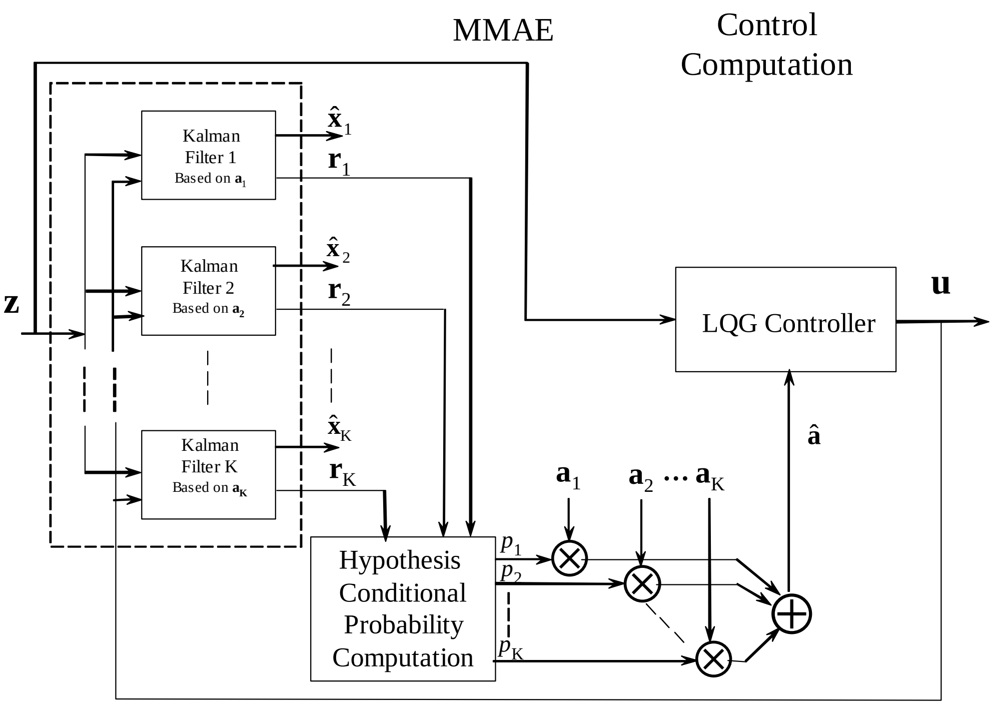

Figure 5.6 MMAE-based control with an LQG controller element

same [56,57] is an indication that the best elemental controller is not generally based on the design for the actual value of the system parameter. Thus the MMAE-based control with an LQG controller element requires a method of discretization of the MMAE and design of the controller that is not necessarily derived directly from the parameter estimate. The generalized MMAC proposed in Chapter 4 is the starting point for the development of this MMAE-based control approach. For the MMAC approach, the full-state feedback elemental controllers associated with the elemental filters in the MMAC are placed in parameter space, but are not necessarily designed using the parameter location of the elemental filter. As with the previous approaches, the design begins with formulation of the performance equations used in the discretization algorithms.

5.3.4.1 Performance Evaluation Equations

The structure of the MMAE-based control with an LQG controller element follows a similar development to that of the generalized MMAC presented in Chapter 4. However, rather than blending elemental LQG controller outputs, the MMAE-based approach developed for this architecture will select one controller. The probability information from the MMAE portion will be used to perform the table look-up. Thus, unlike the previous MMAE-based control scheme, the controller gain *and* the Kalman filter will be *selected* to implement the control element.

Development of the performance evaluation begins with the state equation of the MMAE and then incorporates the elemental LQG. The state equation for the kth filter in the MMAE is given by:

$$\hat{\mathbf{x}}_{\mathrm{k}}(t_{\mathrm{i+1}}^{-}) = \mathbf{\Phi}_{\mathrm{k}}\hat{\mathbf{x}}_{\mathrm{k}}(t_{\mathrm{i}}^{+}) + \mathbf{B}_{\mathrm{d\,k}}\mathbf{u}_{\mathrm{sel}} \tag{5.52}$$

The control in Equation (5.52) is given by the output of the elemental controller expressed as:

$$\mathbf{u}_{\mathrm{sel}}(t_{\mathrm{i}}) = -\mathbf{G}_{\mathrm{c\,sel}}^{*}\hat{\mathbf{x}}_{\mathrm{sel}}(t_{\mathrm{i}}^{+}) \tag{5.53}$$

where superscript *sel* denotes the selected control element.

To derive the equations for the filters in the MMAE, substitute the control from Equation (5.53) into the expression for the filter propagation, Equation (5.52). That substitution results in the expression for the kth elemental filter of the MMAE:

$$\hat{\mathbf{x}}_{\mathrm{k}}(t_{\mathrm{i+1}}^{-}) = \mathbf{\Phi}_{\mathrm{k}}\hat{\mathbf{x}}_{\mathrm{k}}(t_{\mathrm{i}}^{+}) - \mathbf{B}_{\mathrm{d\,k}}\mathbf{G}_{\mathrm{c\,sel}}^{*}\hat{\mathbf{x}}_{\mathrm{sel}}(t_{\mathrm{i}}^{+}) \tag{5.54}$$

The update equation for the kth elemental filter of the MMAE is given by:

$$\hat{\mathbf{x}}_{\mathrm{k}}(t_{\mathrm{i}}^{+}) = \hat{\mathbf{x}}_{\mathrm{k}}(t_{\mathrm{i}}^{-}) + \mathbf{K}_{\mathrm{k}}[\mathbf{z}_{i} - \mathbf{H}_{\mathrm{k}}(t_{i})\hat{\mathbf{x}}_{\mathrm{k}}(t_{\mathrm{i}}^{-})] \tag{5.55}$$

and the update for the Kalman filter in the control element is

$$\hat{\mathbf{x}}_{\text{sel}}(t_{\text{i}}^{+}) = \hat{\mathbf{x}}_{\text{sel}}(t_{\text{i}}^{-}) + \mathbf{K}_{\text{sel}}[\mathbf{z}_i - \mathbf{H}_{\text{sel}}\hat{\mathbf{x}}_{\text{sel}}(t_{\text{i}}^{-})] \tag{5.56}$$

where

$$\mathbf{z}_{\text{i}} = \mathbf{H}_{\text{t}}\mathbf{x}_{\text{t}}(t_{\text{i}}) + \mathbf{v}_{\text{t}}(t_{\text{i}}) \tag{5.57}$$

Now substitute Equations (5.55) and (5.56) into Equation (5.54) to obtain:

$$\begin{aligned}\hat{\mathbf{x}}_{\text{k}}(t_{\text{i+1}}^{-}) = \mathbf{\Phi}_{\text{k}}\Big(\hat{\mathbf{x}}_{\text{k}}(t_{\text{i}}^{-}) + \mathbf{K}_{\text{k}}\big[\mathbf{H}_{\text{t}}\mathbf{x}_{\text{t}}(t_{\text{i}}) + \mathbf{v}_{\text{t}}(t_{\text{i}}) - \mathbf{H}_{\text{k}}\hat{\mathbf{x}}_{\text{k}}(t_{\text{i}}^{-})\big]\Big) \\ - \mathbf{B}_{\text{dk}}\mathbf{G}^{*}_{\text{csel}}\Big(\hat{\mathbf{x}}_{\text{sel}}(t_{\text{i}}^{-}) + \mathbf{K}_{\text{sel}}\big[\mathbf{H}_{\text{t}}\mathbf{x}_{\text{t}}(t_{\text{i}}) + \mathbf{v}_{\text{t}}(t_{\text{i}}) - \mathbf{H}_{\text{sel}}\hat{\mathbf{x}}_{\text{sel}}(t_{\text{i}}^{-})\big]\Big)\end{aligned} \tag{5.58}$$

Simplification of Equation (5.58) yields:

$$\begin{aligned}\hat{\mathbf{x}}_{\text{k}}(t_{\text{i+1}}^{-}) = \mathbf{\Phi}_{\text{k}}(\mathbf{I} - \mathbf{K}_{\text{k}}\mathbf{H}_{\text{k}})\hat{\mathbf{x}}_{\text{k}}(t_{\text{i}}^{-}) + \big(\mathbf{\Phi}_{\text{k}}\mathbf{K}_{\text{k}} - \mathbf{B}_{\text{dk}}\mathbf{G}^{*}_{\text{csel}}\mathbf{K}_{\text{sel}}\big)\mathbf{H}_{\text{t}}\mathbf{x}_{\text{t}}(t_{\text{i}}) \\ - \mathbf{B}_{\text{dk}}\mathbf{G}^{*}_{\text{csel}}(\mathbf{I} - \mathbf{K}_{\text{sel}}\mathbf{H}_{\text{sel}})\hat{\mathbf{x}}_{\text{sel}}(t_{\text{i}}^{-}) + \big(\mathbf{\Phi}_{\text{k}}\mathbf{K}_{\text{k}} - \mathbf{B}_{\text{dk}}\mathbf{G}^{*}_{\text{csel}}\mathbf{K}_{\text{sel}}\big)\mathbf{v}_{\text{t}}(t_{\text{i}})\end{aligned} \tag{5.59}$$

The truth model equations are derived in a similar approach as for the MMAC in Chapter 4. The truth model propagation with the control substitution is given by:

$$\mathbf{x}_{\text{t}}(t_{i+1}) = \mathbf{\Phi}_{\text{t}}\mathbf{x}_{\text{t}}(t_i) - \mathbf{B}_{\text{dt}}\mathbf{G}^{*}_{\text{csel}}\hat{\mathbf{x}}_{\text{sel}}(t_{\text{i}}^{+}) + \mathbf{G}_{\text{dt}}\mathbf{w}_{\text{dt}}(t_i) \tag{5.60}$$

Now substitute Equation (5.56) into Equation (5.60) to yield:

$$\begin{aligned}\mathbf{x}_{\text{t}}(t_{i+1}) = \mathbf{\Phi}_{\text{t}}\mathbf{x}_{\text{t}}(t_i) - \mathbf{B}_{\text{dt}}\mathbf{G}^{*}_{\text{csel}}\Big(\hat{\mathbf{x}}_{\text{sel}}(t_i^{-}) + \mathbf{K}_{\text{sel}}\big[\mathbf{H}_{\text{t}}\mathbf{x}_{\text{t}}(t_i) + \mathbf{v}_{\text{t}}(t_i) - \mathbf{H}_{\text{sel}}\hat{\mathbf{x}}_{\text{sel}}(t_i^{-})\big]\Big) \\ + \mathbf{G}_{\text{dt}}\mathbf{w}_{\text{dt}}(t_i)\end{aligned} \tag{5.61}$$

which simplifies as:

$$\begin{aligned}\mathbf{x}_{\text{t}}(t_{i+1}) = \big(\mathbf{\Phi}_{\text{t}} - \mathbf{B}_{\text{dt}}\mathbf{G}^{*}_{\text{csel}}\mathbf{K}_{\text{sel}}\mathbf{H}_{\text{t}}\big)\mathbf{x}_{\text{t}}(t_i) - \mathbf{B}_{\text{dt}}\mathbf{G}^{*}_{\text{csel}}(\mathbf{I} - \mathbf{K}_{\text{sel}}\mathbf{H}_{\text{sel}})\hat{\mathbf{x}}_{\text{sel}}(t_i^{-}) \\ - \mathbf{B}_{\text{dt}}\mathbf{G}^{*}_{\text{csel}}\mathbf{K}_{\text{sel}}\mathbf{v}_{\text{t}}(t_i) + \mathbf{G}_{\text{dt}}\mathbf{w}_{\text{dt}}(t_i)\end{aligned} \tag{5.62}$$

Now, since the controller element is completely separate from the MMAE, the Kalman filter state description of the LQG controller element must be included in the evaluation. The state estimate is given by:

$$\hat{\mathbf{x}}_{\text{sel}}(t_{\text{i+1}}^{-}) = \mathbf{\Phi}_{\text{sel}}\hat{\mathbf{x}}_{\text{sel}}(t_{\text{i}}^{+}) + \mathbf{B}_{\text{dsel}}\mathbf{u}_{\text{sel}} \tag{5.63}$$

With the control from Equation (5.53), the state estimate for the LQG controller element becomes:

$$\hat{\mathbf{x}}_{\mathrm{sel}}(t_{i+1}^-) = \mathbf{\Phi}_{\mathrm{sel}}\hat{\mathbf{x}}_{\mathrm{sel}}(t_i^+) - \mathbf{B}_{\mathrm{d\,sel}}\mathbf{G}^*_{\mathrm{c\,sel}}\hat{\mathbf{x}}_{\mathrm{sel}}(t_i^+) \tag{5.64}$$

Now substitute the Kalman filter update from control in Equation (5.56) into Equation (5.64), which yields:

$$\hat{\mathbf{x}}_{\mathrm{sel}}(t_{i+1}^-) = \left(\mathbf{\Phi}_{\mathrm{sel}} - \mathbf{B}_{\mathrm{d\,sel}}\mathbf{G}^*_{\mathrm{c\,sel}}\right)\left(\hat{\mathbf{x}}_{\mathrm{sel}}(t_i^-) + \mathbf{K}_{\mathrm{sel}}\left[\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) + \mathbf{v}_{\mathrm{t}}(t_i) - \mathbf{H}_{\mathrm{sel}}\hat{\mathbf{x}}_{\mathrm{sel}}(t_i^-)\right]\right) \tag{5.65}$$

Grouping like terms yields:

$$\begin{aligned}\hat{\mathbf{x}}_{\mathrm{sel}}(t_{i+1}^-) = {} & \left(\mathbf{\Phi}_{\mathrm{sel}} - \mathbf{B}_{\mathrm{d\,sel}}\mathbf{G}^*_{\mathrm{c\,sel}}\right)\mathbf{K}_{\mathrm{sel}}\mathbf{H}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i) + \left(\mathbf{\Phi}_{\mathrm{sel}} - \mathbf{B}_{\mathrm{d\,sel}}\mathbf{G}^*_{\mathrm{c\,sel}}\right)\left(\mathbf{I} - \mathbf{K}_{\mathrm{sel}}\mathbf{H}_{\mathrm{sel}}\right)\hat{\mathbf{x}}_{\mathrm{sel}}(t_i^-) \\ & + \left(\mathbf{\Phi}_{\mathrm{sel}} - \mathbf{B}_{\mathrm{d\,sel}}\mathbf{G}^*_{\mathrm{c\,sel}}\right)\mathbf{K}_{\mathrm{sel}}\mathbf{v}_{\mathrm{t}}(t_i)\end{aligned} \tag{5.66}$$

Now combining all the state equations for the truth model, MMAE, and control yields:

$$\begin{aligned}\begin{bmatrix}\hat{\mathbf{x}}_{\mathrm{k}}(t_{i+1}^-) \\ \mathbf{x}_{\mathrm{t}}(t_{i+1}) \\ \hat{\mathbf{x}}_{\mathrm{sel}}(t_{i+1}^-)\end{bmatrix} = {} & \begin{bmatrix}\mathbf{\Phi}_{\mathrm{k}}\left(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}}\right) & \left(\mathbf{\Phi}_{\mathrm{k}}\mathbf{K}_{\mathrm{k}} - \mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^*_{\mathrm{c\,sel}}\mathbf{K}_{\mathrm{sel}}\right)\mathbf{H}_{\mathrm{t}} & -\mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^*_{\mathrm{c\,sel}}\left(\mathbf{I} - \mathbf{K}_{\mathrm{sel}}\mathbf{H}_{\mathrm{sel}}\right) \\ \mathbf{0} & \left(\mathbf{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c\,sel}}\mathbf{K}_{\mathrm{sel}}\mathbf{H}_{\mathrm{t}}\right) & -\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c\,sel}}\left(\mathbf{I} - \mathbf{K}_{\mathrm{sel}}\mathbf{H}_{\mathrm{sel}}\right) \\ \mathbf{0} & \left(\mathbf{\Phi}_{\mathrm{sel}} - \mathbf{B}_{\mathrm{d\,sel}}\mathbf{G}^*_{\mathrm{c\,sel}}\right)\mathbf{K}_{\mathrm{sel}}\mathbf{H}_{\mathrm{t}} & \left(\mathbf{\Phi}_{\mathrm{sel}} - \mathbf{B}_{\mathrm{d\,sel}}\mathbf{G}^*_{\mathrm{c\,sel}}\right)\left(\mathbf{I} - \mathbf{K}_{\mathrm{sel}}\mathbf{H}_{\mathrm{sel}}\right)\end{bmatrix}\begin{bmatrix}\hat{\mathbf{x}}_{\mathrm{k}}(t_i^-) \\ \mathbf{x}_{\mathrm{t}}(t_i) \\ \hat{\mathbf{x}}_{\mathrm{sel}}(t_i^-)\end{bmatrix} \\ & + \begin{bmatrix}\mathbf{0} & \left(\mathbf{\Phi}_{\mathrm{k}}\mathbf{K}_{\mathrm{k}} - \mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^*_{\mathrm{c\,sel}}\mathbf{K}_{\mathrm{sel}}\right) \\ \mathbf{G}_{\mathrm{d\,t}} & -\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c\,sel}}\mathbf{K}_{\mathrm{sel}} \\ \mathbf{0} & \left(\mathbf{\Phi}_{\mathrm{sel}} - \mathbf{B}_{\mathrm{d\,sel}}\mathbf{G}^*_{\mathrm{c\,sel}}\right)\mathbf{K}_{\mathrm{sel}}\end{bmatrix}\begin{bmatrix}\mathbf{w}_{\mathrm{d\,t}}(t_i) \\ \mathbf{v}_{\mathrm{t}}(t_i)\end{bmatrix}\end{aligned} \tag{5.67}$$

From Equation (5.67) define:

$$\mathbf{T}_{\mathrm{SelC}} = \begin{bmatrix}\mathbf{\Phi}_{\mathrm{k}}\left(\mathbf{I} - \mathbf{K}_{\mathrm{k}}\mathbf{H}_{\mathrm{k}}\right) & \left(\mathbf{\Phi}_{\mathrm{k}}\mathbf{K}_{\mathrm{k}} - \mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^*_{\mathrm{c\,sel}}\mathbf{K}_{\mathrm{sel}}\right)\mathbf{H}_{\mathrm{t}} & -\mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^*_{\mathrm{c\,sel}}\left(\mathbf{I} - \mathbf{K}_{\mathrm{sel}}\mathbf{H}_{\mathrm{sel}}\right) \\ \mathbf{0} & \left(\mathbf{\Phi}_{\mathrm{t}} - \mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c\,sel}}\mathbf{K}_{\mathrm{sel}}\mathbf{H}_{\mathrm{t}}\right) & -\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^*_{\mathrm{c\,sel}}\left(\mathbf{I} - \mathbf{K}_{\mathrm{sel}}\mathbf{H}_{\mathrm{sel}}\right) \\ \mathbf{0} & \left(\mathbf{\Phi}_{\mathrm{sel}} - \mathbf{B}_{\mathrm{d\,sel}}\mathbf{G}^*_{\mathrm{c\,sel}}\right)\mathbf{K}_{\mathrm{sel}}\mathbf{H}_{\mathrm{t}} & \left(\mathbf{\Phi}_{\mathrm{sel}} - \mathbf{B}_{\mathrm{d\,sel}}\mathbf{G}^*_{\mathrm{c\,sel}}\right)\left(\mathbf{I} - \mathbf{K}_{\mathrm{sel}}\mathbf{H}_{\mathrm{sel}}\right)\end{bmatrix} \tag{5.68}$$

and

$$\mathbf{L}_{\mathrm{SelC}} = \begin{bmatrix} \mathbf{0} & \left(\mathbf{\Phi}_{\mathrm{k}}\mathbf{K}_{\mathrm{k}} - \mathbf{B}_{\mathrm{d\,k}}\mathbf{G}^{*}_{\mathrm{c\,sel}}\mathbf{K}_{\mathrm{sel}}\right) \\ \mathbf{G}_{\mathrm{d\,t}} & -\mathbf{B}_{\mathrm{d\,t}}\mathbf{G}^{*}_{\mathrm{c\,sel}}\mathbf{K}_{\mathrm{sel}} \\ \mathbf{0} & \left(\mathbf{\Phi}_{\mathrm{sel}} - \mathbf{B}_{\mathrm{d\,sel}}\mathbf{G}^{*}_{\mathrm{c\,sel}}\right)\mathbf{K}_{\mathrm{sel}} \end{bmatrix} \tag{5.69}$$

The state autocorrelation can now be written as:

$$E\left\{\begin{bmatrix} \hat{\mathbf{x}}_{\mathrm{k}}(t^{-}_{i+1}) \\ \mathbf{x}_{\mathrm{t}}(t_{i+1}) \\ \hat{\mathbf{x}}_{\mathrm{sel}}(t^{-}_{i+1}) \end{bmatrix}\begin{bmatrix} \hat{\mathbf{x}}_{\mathrm{k}}{}^{\mathrm{T}}(t^{-}_{i+1}) & \mathbf{x}_{\mathrm{t}}{}^{\mathrm{T}}(t_{i+1}) & \hat{\mathbf{x}}_{\mathrm{sel}}{}^{\mathrm{T}}(t^{-}_{i+1}) \end{bmatrix}\right\} \\ = \mathbf{\Xi}_{\mathrm{k}}(t^{-}_{i+1}) = \mathbf{T}_{\mathrm{SelC}}\,\mathbf{\Xi}_{\mathrm{k}}(t^{-}_{i})\,\mathbf{T}^{\mathrm{T}}_{\mathrm{SelC}} + \mathbf{L}_{\mathrm{SelC}}\mathbf{Q}_{\mathbf{0}}\mathbf{L}^{\mathrm{T}}_{\mathrm{SelC}} \tag{5.70}$$

The output autocorrelation is given by $\mathrm{E}\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\,\mathbf{y}\} \equiv \mathrm{tr}\left(\mathbf{W}\,\mathbf{C}\,\mathbf{\Psi}_{\mathbf{x}_{\mathrm{t}}}\mathbf{C}^{\mathrm{T}}\right)$ and $\mathbf{\Psi}_{\mathbf{x}_{\mathrm{t}}} = \mathrm{E}\{[\mathbf{x}_{\mathrm{t}}][\mathbf{x}_{\mathrm{t}}]^{\mathrm{T}}\}$.

It is clear from Equation (5.67) that the MMAE portion of the architecture does not *directly* affect the performance. However, Equation (5.67) does not show how the selection of the controller is dependent on the parameter estimates from the MMAE. Since the MMAE is dependent on the elemental controller, that will affect the placement of the filters in parameter space.

The only way to tie the design of the selected LQG controller to the MMAE is through the parameter estimate. As discussed for the previous MMAE-based control approaches, more than one set of probabilities can define the same parameter estimate. Thus, the probabilities are used as the table look-up index since the state estimation for the control was derived using those probabilities. The probability blending to derive the state estimate allows the performance measure to be expressed in terms of the probabilities. As discussed in the next section, since the state estimation for the LQG control element is formed separately from that of the MMAE, the performance measure cannot be written in terms of the probabilities.

5.3.4.2 Probability Lower Bounding

For the MMAE-based control with the LQG controller element, the impact of a probability lower bound is different from that of the previous MMAE-based approaches. The difference comes from the observation that no portion of the control is directly derived from the MMAE. In the previous cases, the probabilities form the state estimates as well as serve as the means to select the controller, either directly or indirectly. Thus, there was a direct link between the probabilities or the parameter estimate of the MMAE portion and the controller performance.

Lower bounding for this architecture does not prevent controller lockout insomuch as it prevents the lockout in the MMAE that chooses the controller. Hence, it is still desirable to discretize the parameter space with the intent on implementing lower bounding. The modifications to the performance measure will be to the state and parameter estimation for the MMAE portion. Thus, rather than assuming that one controller will be selected with a probability of one at steady state and the rest zero (as with no lower bound imposed on computed probabilities), all the filters will have a probability of $p_{\min}$ except for one filter. That filter will have a probability expressed as:

$$p_{\text{sel}} = 1 - p_{\min}(\text{K-1}) \tag{5.71}$$

So, in terms of the probabilities from the MMAE portion, the state estimate will be:

$$\hat{\mathbf{x}} = (1 - p_{\min}\text{K})\hat{\mathbf{x}}_{\text{k}} + \sum_{\text{j=1}}^{\text{K}} p_{\min}\hat{\mathbf{x}}_{\text{j}} \tag{5.72}$$

and the corresponding parameter estimate is similar:

$$\hat{\mathbf{a}} = (1 - p_{\min}\text{K})\hat{\mathbf{a}}_{\text{k}} + \sum_{\text{j=1}}^{\text{K}} p_{\min}\hat{\mathbf{a}}_{\text{j}} \tag{5.73}$$

To determine the parameter estimate at steady state that will be used to select the controller, the state equations need to reflect that, at steady state, the probability will be assigned according to Equation (5.71) and the discussion preceding it. The parameter estimate used to select the controller is evaluated according to Equation (5.73).

The state estimate outputs from the MMAE elemental filters are not used in the control computation and thus Equation (5.72) would not need to be computed for the performance measure. The elemental LQG controller, of course, has its own state estimate. The filters in the MMAE can be evaluated according to Equation (5.67). However, the key is that the controller $G^{*}_{c_{sel}}$ that is used in the evaluation of Equation (5.67) is selected using Equation (5.73). Hence, the arbitrary lower bound is reflected in the performance evaluation through the parameter estimate and the corresponding controller gain. The gain selection will inevitably affect the discretization of the parameter space.

5.3.4.3 Discretization Algorithm

There are two requirements for the discretization algorithm for the MMAE-based control with an LQG controller element. The first is the optimal placement of the filters in the assigned parameter space for the MMAE portion of the architecture. The second is the specification of the look-up table for the elemental LQG controllers.

In order to evaluate the performance measure, the selected controller has to be determined, which of course is determined by the MMAE. As in the MMAC [56,57], the filter that is assumed to have the maximum probability at steady state is the filter with the minimum of the proximity measure given by:

$$\ell \equiv \min \ell_k \qquad k = 1,\ldots K \tag{5.74}$$

where

$$\ell_k \equiv \log|\mathbf{A}_k| + \mathrm{tr}[\mathbf{A}_k^{-1}\mathbf{N}_k] \tag{5.75}$$

$\mathbf{A}_k$ is the covariance of the steady state residuals in the kth filter, i.e., $\left[\mathbf{H}_k\,\mathbf{P}^-_{k\infty}\,\mathbf{H}_k^T + \mathbf{R}_k\right]$.

$\mathbf{N}_k$ is the actual steady state autocorrelation of the estimation errors in the kth filter and is given by:

$$\mathbf{N}_k = \left[-\mathbf{H}_k \quad \mathbf{H}_t \quad \mathbf{0}\right] \boldsymbol{\Xi}_{k\infty} \left[-\mathbf{H}_k \quad \mathbf{H}_t \quad \mathbf{0}\right]^T \tag{5.76}$$

where $\boldsymbol{\Xi}_{k\infty}$ is the state prediction autocorrelation computed by Equation (5.70) and $\mathbf{H}_k$ and $\mathbf{H}_t$ are the output matrix of the kth filter of the MMAE and truth model, respectively.

As evident in Equation (5.67), the control must be available for the state prediction autocorrelation for the MMAE, which in turn will be used to select the controller according to Equation (5.74). Thus, it is necessary to determine the best controller for any point in parameter space. The design of the generalized controller will use the LQG techniques covered in Chapter 3.

Determination of the filter at steady state in the MMAE and design of the best controller for the assumed steady state parameter estimate are two components necessary for the discretization algorithm for the MMAE. Using these two techniques, the discretization algorithm can be specified. It follows similar steps as the generalized MMAC from Chapter 4 which are summarized as follows:

1. Describe in terms of the parameter vector **a**, the truth model of the system, the filter in the MMAE and the LQG controller element.
2. Choose the number of filters K in the MMAE.

3. Choose a representative parameter set, $\{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_K\}$ to begin the minimization.

4. For each parameter in the representative parameter set $\{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_K\}$, design a generalized LQG elemental controller.

5. Use a numerical integration technique to evaluate the cost function given by:

$$J_{2\ \text{c-ss}} \equiv \frac{\int_A E\{\mathbf{y}^T \mathbf{W} \mathbf{y}\} da}{\int_A da} \tag{5.77}$$

where $E\{\mathbf{y}^T \mathbf{W}\,\mathbf{y}\} \equiv \text{tr}(\mathbf{W}\,\mathbf{C}\,\boldsymbol{\Psi}_{\mathbf{x}_t} \mathbf{C}^T)$ and $\boldsymbol{\Psi}_{\mathbf{x}_t} = E\{[\mathbf{x}_t][\mathbf{x}_t]^T\}$.

   a. Compute $\boldsymbol{\Xi}_{k_\infty}$ using Equation (5.70) at discrete points in the parameter space (for $\mathbf{a}_t$ value) for each filter: k = 1,…K and the corresponding generalized LQG controller element.

   b. At each discrete point, evaluate Equation (5.74), the proximity measure, using $\boldsymbol{\Xi}_{k_\infty}$ for k = 1,…K to determine the convergence to a single filter.

   c. For the selected elemental filter, determine $\boldsymbol{\Psi}_{\mathbf{x}_t}$ from the previous evaluation of $\boldsymbol{\Xi}_{k_\infty}$ for k = 1,…K and use in the evaluation of the cost function.

2) Use a vector minimization technique with the functional evaluation from Step (5) to minimize $J_{2\ \text{c-ss}}$.

As is discussed for the previous MMAE-based control look-up approaches, the steady state discretization only provides K (the number of filters in the MMAE) entries in the controller look-up table. However, the size of the look-up table is only limited by the resource constraints of the physical implementation. For this approach, it has been discussed that the parameter estimate from the MMAE portion will serve as the index into the table of LQG controllers. Thus, discrete points in parameter space are used to design generalized LQG controllers as the entries in the table. This of course assumes that MMAE provides accurate parameter estimates.

5.3.4.4 Discussion

The MMAE-Based Control with an LQG Controller Element is the final level of refinement for the MMAE-based control. In fact, this architecture can be considered the most general case of MMAE-based control. At steady state, the previous architectures can be expressed in terms of the MMAE-based control with an LQG controller element. For this architecture to be the same as the previous architectures at steady state, the former must have the Kalman filter in the LQG portion match the filter in the MMAE portion (the filter selected at steady state). Additionally, the LQG control gain matrix also has to match those in the previous architectures. For example, to match the MMAE-based control with a nominal controller, all the gain matrices in the LQG controller look-up table have to be equivalent to the nominal controller gain matrix. During the transient phase, the MMAE-based control with an LQG controller element cannot be reduced to the same form as the previous MMAE-based architectures. In the transient phase of the control response, the previous MMAE-based architectures blend multiple state estimates using the probability vector as weights. This cannot be duplicated exactly with the single Kalman filter in the LQG controller. Hence, a direct comparison of the architectures cannot be made for the time during the transient response.

Since the MMAE-based control with an LQG controller element parallels the implementation for the GMMAC, the steady state performance will be the same. The transient performance generally will be different. Design of a single LQG controller based on a single design point will not be equivalent to the control from blending several controllers based on different design points as is done in the MMAC architecture. Comparisons of the transient performance will have to be accomplished using Monte Carlo simulations.

The MMAE-based control with an LQG controller element requires more resources than a conventional MMAE-based control system. Primarily, for the table look-up approach, rather than just storing a gain matrix to be indexed, the Kalman filter must be stored as well. There is also the issue of computing the extra Kalman filter as well. For the MMAC, the MMAE substructure and the LQG controller element, if the steady state Kalman filter gain is used (as will be the case for the implementation in Chapter 6), the complexity is reduced to the state propagation and update equations. However, determining the probability for the MMAC and MMAE estimation outputs requires the additional conditional probability density function computations. This is considerably greater than the computations strictly required for the filters. Hence, the complexity versus performance issue is not the same as adding one filter to the MMAE substructure or the MMAC to try to improve performance.

5.4 Summary

This chapter advances the MMAE-based architecture beyond its rudimentary form of a state estimate from an MMAE multiplied by a gain factor (evaluated adaptively or not). In some instances, the proposed architectures build upon concepts from the enhancements proposed for the MMAC. The development begins with discussion of the MMAE-based control with a nominal controller and goes on to propose more versatile and general forms of MMAE-based control. Regardless of the architectures, the goal was to present procedures for an optimal implementation. This advances the design of the MMAE-based control beyond ad hoc design procedures.

Of primary importance in this chapter was the analysis of the conventional MMAE-based structure, especially in comparison to the MMAC architecture. The analysis demonstrated that the two architectures are identical at steady state. This enabled similar design approaches developed for the MMAC to be applied to the MMAE-based control architecture. The steady state performance evaluation determines how to place the filters in the MMAE and of course the corresponding controller element. For the table look-up and LQG controller element approaches, additional discretization was developed to determine the controllers that are referenced by the MMAE.

Chapter 6 - Application Evaluation

This chapter applies the design approaches developed in the previous sections to a sample problem. Rather than attempt to use a complex or perhaps a real-world application, a two-state system in which a single parameter is allowed to vary was chosen. Though this is a simple problem, it is representative of real-world applications such as first order bending mode control, which has been the subject of previous multiple model research [16,17,18,19,20,52,53,54]. This two state problem is the same as what Sheldon studied [56], as was also done by Hentz [22]. Since this work builds upon and expands Sheldon's approach to MMAC design, it is reasonable to use the same example to compare results. In addition, since this work also investigates MMAE-based control, this example will provide a means to compare the different multiple model techniques as perhaps never previously accomplished. Finally, a simple system with a single uncertain parameter reduces the complexity of analysis. Thus, conclusions drawn in this analysis are free from potential interdependency of multiple parameters that affects the results rather than the veracity of the technique.

6.1 System Description

The ideal mechanical translational system as shown in Figure 6.1 is a continuous-time system second order system of the form

$$\dot{\mathbf{x}}(t) = \mathbf{F}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t) \tag{6.1}$$

modeled as

$$\begin{bmatrix} \dot{\mathrm{x}}_1(t) \\ \dot{\mathrm{x}}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{b}{m} \end{bmatrix} \begin{bmatrix} \mathrm{x}_1(t) \\ \mathrm{x}_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{k}{m} \end{bmatrix} u(t) \tag{6.2}$$

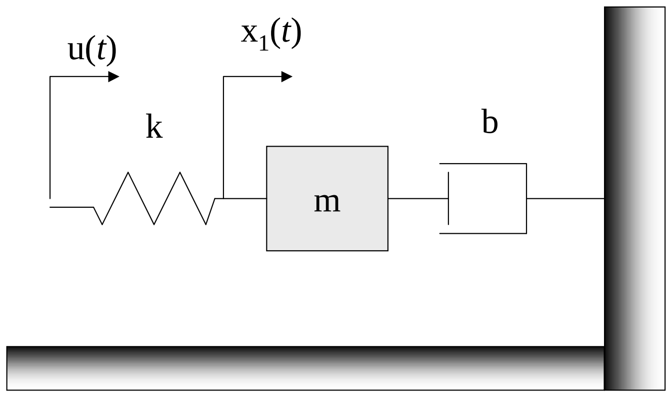

Figure 6.1 Ideal mechanical translational system

where x_1 is the position and x_2 is the velocity. The constants k, m, and b are the spring constant, mass, and damping coefficient, respectively. This true system model can be simplified into the more general form by assigning the undamped natural frequency as $\omega_n \equiv \sqrt{\frac{k}{m}}$ and letting the damping ratio be $\zeta \equiv 2\frac{b}{\sqrt{km}}$. Adding dynamic driving noise yields the stochastic truth model:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_n^2 & -2\zeta\omega \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \omega_n^2 \end{bmatrix} u(t) + \begin{bmatrix} 0 \\ \omega_n^2 \end{bmatrix} w(t) \tag{6.3}$$

with $w(t)$ being zero-mean white Gaussian noise of strength Q:

$$\mathrm{E}\{w(t)w(t+\tau)\} = Q\delta(\tau)$$

For the purpose of computer simulation and control, the system model given in Equation (6.1) is converted to an equivalent discrete model via [34]:

$$\mathbf{x}(t_{i+1}) = \mathbf{\Phi}(t_{i+1}, t_i)\mathbf{x}(t_i) + \mathbf{B}_d(t_i)\mathbf{u}(t_i) + \mathbf{w}_d(t_i) \tag{6.4}$$

where the sample period t_{i+1}-t_i = 0.01 sec, $\mathbf{\Phi}(t_{i+1}, t_i)$ is the state transition matrix equal to $e^{\mathbf{F}\Delta t}$ since **F** is constant for this model, and

$$\mathbf{B}_\mathrm{d}(t_i) = \int_{t_i}^{t_{i+1}} \mathbf{\Phi}(t_{i+1}, \tau)\mathbf{B}(\tau)d\tau\,. \tag{6.5}$$

The discrete-time white Gausian noise term $\mathbf{w}_d(t_i)$ has zero mean and covariance kernel:

$$\mathrm{E}\{\mathbf{w}_d(t_i)\mathbf{w}_d(t_j)^{\mathrm{T}}\} = \begin{cases} \mathbf{Q}_{\mathrm{d}}(t_i) & t_i = t_j \\ \mathbf{0} & t_i \neq t_j \end{cases} \tag{6.6}$$

where

$$\mathbf{Q}_{\mathrm{d}}(t_i) = \int_{t_i}^{t_{i+1}} \mathbf{\Phi}(t_{i+1}, \boldsymbol{\tau})\mathbf{G}(\boldsymbol{\tau})Q(\boldsymbol{\tau})\mathbf{G}^{\mathrm{T}}(\boldsymbol{\tau})\mathbf{\Phi}^{\mathrm{T}}(t_{i+1}, \boldsymbol{\tau})d\boldsymbol{\tau} \tag{6.7}$$

Measurements are taken at each sample time and described by

$$\mathrm{z}(t_i) = \mathbf{H}(t_i)\mathbf{x}(t_i) + \mathrm{v}(t_i) \tag{6.8}$$

where **H** is the output measurement matrix. $\mathrm{v}(t_i)$ is a zero-mean white Gaussian noise process with covariance kernel

$$\mathrm{E}\{\mathrm{v}(t_i)\mathrm{v}(t_j)^{\mathrm{T}}\} = \begin{cases} \mathrm{R}(t_i) & t_i = t_j \\ 0 & t_i \neq t_j \end{cases} \tag{6.9}$$

R is the variance of the measurement noise. Finally, the output to be controlled is given by

$$\mathrm{y}(t_i) = \mathbf{C}(t_i)\mathbf{x}(t_i) \tag{6.10}$$

where **C** is the output matrix.

6.2 Evaluation Details

This section defines the scope of the general system description of the previous section and outlines the details of the problem analysis. Limiting the unknown factors allows a framework so that clear and meaningful conclusions on performance can be made. The sample problem will be used to compare LQG design approaches as well as the proposed multiple model structures.

6.2.1 Problem Assumptions

It is apparent from the system description in the previous section that there are potentially four unknowns: the undamped natural frequency ω_n, the damping ratio ζ, the dynamic driving noise strength Q, and the variance of the measurement noise R. This example will make the same simplification as Sheldon did and keep the latter three components constant and assign them the following values:

$$Q = 0.01$$

$$\zeta = 0.01$$

$$R(t_i) = 0.01 \ \forall\, i$$

As illustrated earlier, this simple two state problem has real-world application. Here, the light damping is representative of bending modes in flexible structures as found in many space vehicles.

For this application, the undamped natural frequency is considered the uncertain parameter. This parameter is restricted to the following range:

$$2\pi \frac{\text{rad}}{\text{sec}} < \omega_n < 20\pi \frac{\text{rad}}{\text{sec}} \tag{6.11}$$

Restricting the range of the uncertain parameter space is equivalent to bounding a problem to realistic operating ranges in a real-world implementation. Any deviations outside the given acceptable range could be considered a system failure, and failure detection is not in the scope of this work.

The system state propagation matrix **F**, the control input matrix **B**, and input noise matrix **G** are determined by the undamped natural frequency, as seen in Equation (6.3). The remaining measurement matrix and output matrix are defined as:

$$\mathbf{H} = [1 \quad 0]$$

$$\mathbf{C} = [1 \quad 0]$$

In other words, for this experiment, only the position will be measured, and that position will be considered the controlled variable for the synthesis of controllers.

6.2.2 Evaluation Approach

There are three categories of design approaches studied in the previous chapters that will be evaluated: LQG design, MMAC and MMAE-based control. For the first, LQG, the typical design approach will be evaluated and will be considered the baseline case. The proposed advances to LQG design will be evaluated and compared with that baseline case. For the MMAC, Sheldon's results, duplicated here, will be considered the baseline case. Therefore, the proposed advances to MMAC design will be compared with Sheldon's baseline results as well as with the nonadaptive LQG design results. Finally, the MMAE-based control design approach really does not have an approach that could truly be considered a baseline. The MMAC design will be considered the benchmark which to compare the MMAE-based control.

Though the primary purpose of performing the evaluation of the design approaches is to demonstrate improvements to LQG and multiple model designs, the secondary purpose is to show the effectiveness of the performance prediction tools that were developed in Chapter 3 through Chapter 5. Thus, the performance will be evaluated with the prediction tools and those results will be compared with Monte Carlo evaluation. Both the prediction evaluation and the Monte Carlo evaluation will be used to compare design techniques and implementation approaches.

The evaluation of performance of each controller implementation will be over the parameter space given in Equation (6.11). In reality, this is a continuous parameter space. For simulation and computer evaluation, this parameter space is discretized into 200 evenly sampled points, where each point corresponds to the possible true system. At each parameter point, the predicted output (mean squared value) and mean (of the output squared) and standard deviation of a 20-run Monte Carlo simulation will be computed and plotted. This data also will be used to compare the performances of the design approaches over the defined parameter space. To determine relative performance of each design, the average (averaged over the parameter space) of the predicted mean squared output and the average of the squared output of the Monte Carlo simulations over the 200-point parameter space are computed and compared.

An evaluation of performance is determined similarly to the performance measure used for discretization of the parameter space. A meaningful scalar measure of the performance can be written as:

$$\text{Predicted Performance Measure} = \mathrm{E}\{\mathbf{y}_t^{\mathrm{T}}\mathbf{W}\mathbf{y}_t + \mathbf{u}^{\mathrm{T}}\mathbf{U}\mathbf{u}\} \qquad (6.12)$$

Further, it is useful to consider the performance of each, the output and the control, individually. Consider first the scalar value that gives an evaluation of performance of the output and is expressed as:

$$\begin{array}{c}\text{Predicted Output}\\ \text{Regulation Error Measure}\end{array} = \mathrm{E}\{\mathbf{y}_t^{\mathrm{T}}\mathbf{W}\mathbf{y}_t\} \qquad (6.13)$$

This scalar evaluation can be derived from the output correlation as:

$$\mathrm{E}\{\mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{y}\} = \mathrm{tr}(\mathbf{W}\mathrm{E}\{\mathbf{y}_t\mathbf{y}_t^{\mathrm{T}}\}) \qquad (6.14)$$

Now, substitute the expression for the output, $\mathbf{y}_t = \mathbf{C}_t\mathbf{x}_t$ and the predicted output regulation error in terms of the predicted true state autocorrelation becomes:

$$\mathrm{E}\{\mathbf{y}_t^{\mathrm{T}}\mathbf{W}\mathbf{y}_t\} = \mathrm{tr}(\mathbf{W}\mathbf{C}_t\ \mathrm{E}\{\mathbf{x}_t\mathbf{x}_t^{\mathrm{T}}\}\mathbf{C}_t^{\mathrm{T}}) \qquad (6.15)$$

Note that Equation (6.15) uses the autocorrelation of the augmented state vector from:

$$E\left\{\begin{bmatrix}\mathbf{x}_{\mathrm{t}}(t_i)\\ \hat{\mathbf{x}}(t_i^-)\end{bmatrix}\begin{bmatrix}\mathbf{x}_{\mathrm{t}}(t_i)^{\mathrm{T}} & \hat{\mathbf{x}}(t_i^-)^{\mathrm{T}}\end{bmatrix}\right\}$$

which can be evaluated either analytically or by Monte Carlo methods. This autocorrelation is the basis for all performance measures used in this chapter.

Equation (6.15) is evaluated at each point in the parameter space and thus the average (over the parameter space) of the regulation output error is given as:

$$\begin{array}{c}\text{Average Predicted}\\ \text{Output Regulation}\\ \text{Error Measure}\end{array} = \frac{1}{(200)}\sum_{\mathrm{k}=1}^{200}\mathrm{tr}(\mathbf{W}\mathbf{C}_{\mathrm{t}}\ \mathrm{E}\{\mathbf{x}_{\mathrm{t\,k}}(t_\infty)\mathbf{x}_{\mathrm{t\,k}}(t_\infty)^{\mathrm{T}}\}\mathbf{C}_{\mathrm{t}}^{\mathrm{T}}) \qquad (6.16)$$

which is the most general form. For this example, the analytically derived predicted measures are being evaluated at steady-state, hence time is taken at t_∞. It is also possible to use a time-averaged value of state over some interval of interest such as a transient period, rather than a value at what is assumed to be steady state.

Now since the output matrix is defined as $\mathbf{C}_{\mathrm{t}}$ = [1 0] for the example being considered, only the position correlation is considered. In addition, to be consistent with the design process, the weighting matrix is selected as $\mathbf{W}=\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}$. Thus, the average of the position regulation error becomes:

$$\begin{array}{c}\text{Average Predicted}\\ \text{Position Regulation}\\ \text{Error}\end{array} = \frac{1}{(200)}\sum_{\mathrm{k}=1}^{200}\mathrm{E}\{\mathrm{x}_{1_{\mathrm{t\,k}}}(t_\infty)\mathrm{x}_{1_{\mathrm{t\,k}}}(t_\infty)\} \qquad (6.17)$$

The predicted value for the quadratic on control from Equation (6.12) is derived in a similar manner as the output. The predicted control quadratic is expressed as:

$$\begin{matrix}\text{Predicted Control}\\ \text{Quadratic}\end{matrix} = \mathrm{E}\{\mathbf{u}^{\mathrm{T}}\mathbf{U}\mathbf{u}\} \tag{6.18}$$

This scalar evaluation can be derived from the control autocorrelation as:

$$\mathrm{E}\{\mathbf{u}^{\mathrm{T}}\mathbf{U}\mathbf{u}\} = \mathrm{tr}(\mathbf{U}\mathrm{E}\{\mathbf{u}\mathbf{u}^{\mathrm{T}}\}) \tag{6.19}$$

Now, substitute the expression for the control, $\mathbf{u} = -\mathbf{G}_{\mathrm{c}}^{*}\hat{\mathbf{x}}$ and the predicted control quadratic in terms of the state estimation autocorrelation becomes:

$$\mathrm{E}\{\mathbf{u}^{\mathrm{T}}\mathbf{U}\mathbf{u}\} = \mathrm{tr}(\mathbf{U}\mathbf{G}_{\mathrm{c}}^{*}\,\mathrm{E}\{\hat{\mathbf{x}}\hat{\mathbf{x}}^{\mathrm{T}}\}\,\mathbf{G}_{\mathrm{c}}^{*\mathrm{T}}) \tag{6.20}$$

Recall that $\mathrm{E}\{\hat{\mathbf{x}}\ \hat{\mathbf{x}}^{\mathrm{T}}\}$ comes from the autocorrelation of the augmented state vector as previously discussed. Equation (6.20) is evaluated at each point in the parameter space and thus the average (over the parameter space) of the control quadratic is given as:

$$\begin{matrix}\text{Average Predicted}\\ \text{Control Quadratic}\end{matrix} = \frac{1}{(200)}\sum_{\mathrm{k}=1}^{200}\mathrm{tr}(\mathbf{U}\mathbf{G}_{\mathrm{c}}^{*}\,\mathrm{E}\{\hat{\mathbf{x}}_{\mathrm{k}}(t_{\infty})\hat{\mathbf{x}}_{\mathrm{k}}(t_{\infty})^{\mathrm{T}}\}\mathbf{G}_{\mathrm{c}}^{*\mathrm{T}}) \tag{6.21}$$

which is the most general form.

The approach to computing the mean squared output for the Monte Carlo evaluation is very similar to Equation (6.16). The length of run for each the simulations is 10 seconds and it is assumed that steady state is reached within 2 seconds. The output is averaged over the subsequent 8 seconds or 800 points to yield the time-averaged steady state regulation output error. Thus, for each point (k) in the parameter space, the mean squared steady-state regulation output is computed as:

$$\text{Mean Squared Output Measure}\,(k) = \frac{1}{20}\sum_{\mathrm{run}=1}^{20}\left\{\frac{1}{800}\sum_{i=201}^{1000}\mathbf{y}_{\mathrm{run}}(t_i)^{\mathrm{T}}\mathbf{y}_{\mathrm{run}}(t_i)\right\} \tag{6.22}$$

where $\mathbf{y}_{\mathrm{t}}(t_i) = \mathbf{C}_{\mathrm{t}}\mathbf{x}_{\mathrm{t}}(t_i)$. Now the average mean squared output over the parameter space for the problem is defined as:

$$\begin{array}{c}\text{Average Mean Squared}\\ \text{Output Measure}\end{array} = \frac{1}{200}\sum_{\mathrm{k}=1}^{200}\text{Mean Squared Output (k)} \qquad (6.23)$$

where k is the index into the parameter space corresponding to the mean squared output computed at that point. Again, since the output matrix is defined as $\mathbf{C}_{\mathrm{t}}$ = [1 0], only the mean squared position regulation error is considered, the spatially averaged mean squared steady-state position is defined as:

$$\begin{array}{c}\text{Average Mean Squared Position}\\ \text{Regulation Error Measure}\end{array} = \frac{1}{200}\sum_{\mathrm{k}=1}^{200}\frac{1}{20}\sum_{\mathrm{run}=1}^{20}\left\{\frac{1}{800}\sum_{i=201}^{1000}\left[x_{1\,\mathrm{k}}(t_i)\right]^2\right\} \qquad (6.24)$$

The final performance measure defined is the mean squared measure for the control for the Monte Carlo simulations. It is computed in a very similar manner as the average mean squared output regulation error. For each point (k) in the parameter space, the mean squared steady-state control is computed as:

$$\text{Mean Squared Control Measure (k)} = \frac{1}{20}\sum_{\mathrm{run}=1}^{20}\left\{\frac{1}{800}\sum_{i=201}^{1000}\mathbf{u}_{\mathrm{run}}(t_i)^{\mathrm{T}}\mathbf{u}_{\mathrm{run}}(t_i)\right\} \qquad (6.25)$$

where $\mathbf{u}(t_i) = -\mathbf{G}_{\mathrm{c}}\hat{\mathbf{x}}(t_i)$. Now the average mean squared control over the parameter space for the problem is defined as:

$$\begin{array}{c}\text{Average Mean Squared}\\ \text{Control Measure}\end{array} = \frac{1}{200}\sum_{\mathrm{k}=1}^{200}\text{Mean Squared Control (k)} \qquad (6.26)$$

6.3 LQG Evaluation

This section investigates the application and evaluates the performance of the LQG controller optimization algorithms developed in Chapter 3 to the ideal mechanical-translational system defined in Section 6.1. The first step in the evaluation process is to establish a baseline of performance using the typical LQG controller design approach. The next step tests the modified LQG (MLQG) design developed in Section 3.2 under the same conditions as the baseline case and compares the results. The final evaluation defines and tests the robustness of the LQG controller designs. The subsequent discussion compares the performance of the typical LQG controller and the MLQG controller with enhanced robustness developed in Section 3.3.

Rather than apply the design algorithms at a single parameter point or assumed set of system parameters, the tests for the LQG evaluation use the same parameter space as for the multiple model controller testing and evaluation. The number of discrete points in the parameter space was selected arbitrarily to be 200, but it was found that additional models did not significantly impact the computed average over the parameter space of the mean squared position error. To state the LQG evaluation in terms of multiple model control, it is artificially assumed that a parameter estimator returns a perfect estimate of the uncertain parameter of the system, in this case, the undamped natural frequency. Now, using one of the proposed design techniques, an LQG controller is designed using the system model, with that assumed value for the parameter. The position correlation (mean squared regulation error) can be calculated for that design and system model, from which a meaningful scalar is computed to be used to evaluate performance. Now this process is repeated for each discrete point in the parameter space. Thus, the performance of LQG controller designs can be compared against that of the multiple model controller designs and will be the benchmark for all performance comparisons. Prediction of the performance of the multiple model controllers versus the typical LQG controllers over the same parameter space will be made in the subsequent sections that discuss the multiple model controller tests.

6.3.1 LQG Baseline Case

Use of the conventional LQG design approach applied at each point in the parameter space defines the baseline case for the LQG analysis. It is assumed that, at each point in parameter space, the system is known completely. Conventional design techniques are assumed to be the best approach for design of an LQG controller. For the conventional approach, the system model for the design uses the assumed parameter at each point in parameter space. The analysis will evaluate the predicted position correlation (mean squared position) at each point in parameter space as well as calculate the cost. Monte Carlo analysis will be used to verify the predictive analysis.

6.3.1.1 Baseline Case Controller Design

As used throughout this application evaluation, the design method used for the baseline case is the full-state feedback LQG controller in which the state estimates are from a Kalman filter and the gains are designed to minimize to quadratic functional of the form:

$$J = \int_0^\infty [\mathbf{x}^{\mathrm{T}}(t)\mathbf{W}_{\mathbf{xx}}\mathbf{x}(t) + \mathbf{u}^{\mathrm{T}}(t)\mathbf{W}_{\mathbf{uu}}\mathbf{u}(t)]dt \qquad (6.27)$$

The weights $\mathbf{W}_{\mathbf{xx}}$ and $\mathbf{W}_{\mathbf{uu}}$ are of course design parameters to allow the emphasis to be placed on the desired vector element quantities. For emphasis on position without regard to the amount of control (i.e., the so called *cheap* control case), the following weights are assigned:

$$\mathbf{W}_{\mathbf{xx}} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

$$\mathbf{W}_{\mathbf{uu}} = [0.0001]$$

The very small non-zero weight on control (rather than zero explicitly) prevents computation errors due to the limitations of the MATLAB implementation.

Besides minimizing the quadratic cost functional in Equation (6.27), the typical LQG design technique assumes that the states in the cost functional are defined by the same model as that for the system. Thus, for the baseline case, the system model and filter and controller design models are equivalent. Figure 6.2 illustrates that the filter models and controller models are specified by the parameter that also corresponds to the point at which the system is based. Since, for the baseline case, the controller and filter models are the same as the system model, the plots are coincident with a slope of one.

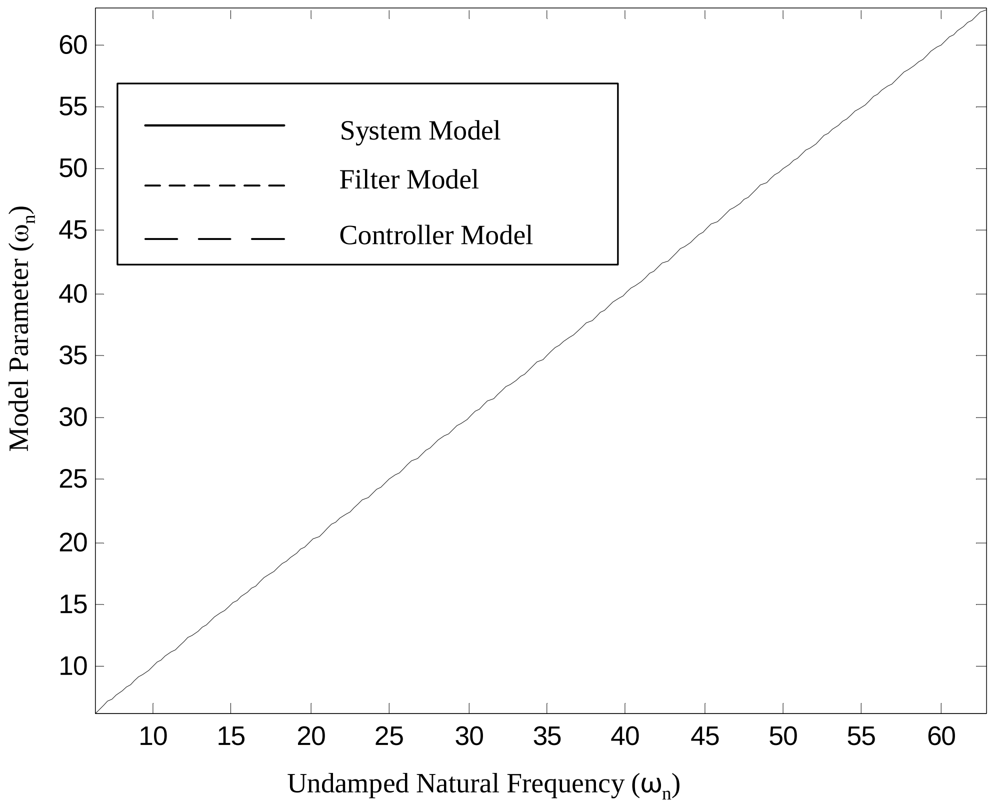

Figure 6.2 System, filter, and controller models used in LQG design

6.3.1.2 Baseline Case Controller Results

For the baseline case, analysis uses the results of both the predicted mean squared output regulation error computation and a 20-run Monte Carlo simulation. For the Monte Carlo simulation, as indicated in the introduction, the duration of the run is 10 seconds with the last 8 seconds assumed to be steady state. Figure 6.3 is the plot of the simulation results overlaid with the predicted output. Clearly, the predicted output is indicative of simulation results. As Table 6.1 indicates, there is a negligible .06% difference between predicted performance and the Monte Carlo simulation for the mean squared position measure averaged over the parameter space. Clearly, the predictive measure is a good representation of the performance over the parameter space.

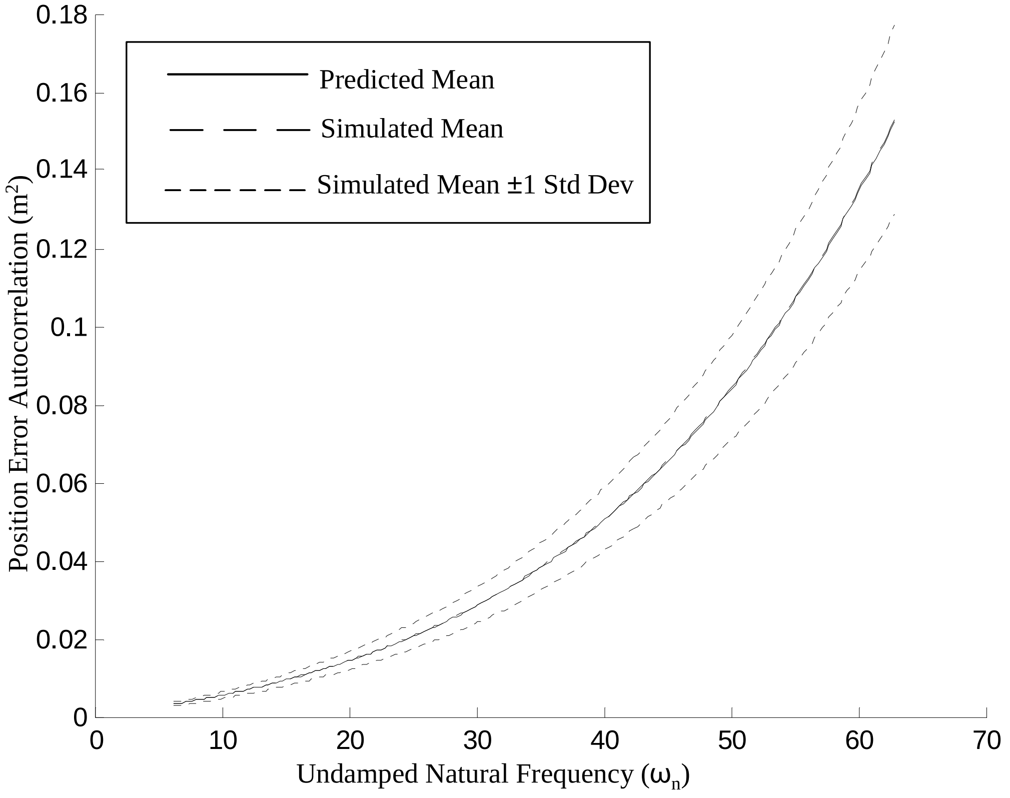

Figure 6.3 Performance of the LQG controller for the parameter space.

	Predictive Measure Average Mean Squared Position (Meters^2)	Monte Carlo Simulation Average Mean Squared Position (Meters^2)
LQG Controller	.0503	.0506

Table 6.1 Results for the LQG controller analysis

6.3.2 Modified LQG Controller

This section applies the techniques for the MLQG controller designs described in Section 3.2 to the ideal mechanical-translational system defined in Section 6.1. There were three different approaches proposed for the filter and controller selection to accomplish the MLQG design. In summary, these techniques attempt to improve on the conventional design by allowing the optimization algorithm to *find* the value of the parameter for the model that represents the optimal filter, the optimal controller, or optimal filter and controller for the best control performance in terms of the anticipated mean square position regulation error averaged over the parameter space. As described in the first section, the design model parameter is the natural frequency, ω_n.

As stated in Chapter 3, if the insight from Luenberger [23] is applied, then it is expected that any improvements would come from a filter design that is based on a model that is *faster* than the system model. Thus, for this example problem, it is anticipated that, for the design in which ω_n for the controller model is the same as for the actual system model, the filter will be based upon an ω_n that is greater than the ω_n in both the controller and the system. Further, when ω_n for the filter is fixed to be same as for the system model, the controller model also will be the same as the system model or a model equivalent to a larger controller gain. A larger gain will correspond to a controller model with a smaller ω_n. A similar prediction is extended to the generalized MLQG. The filter model will correspond to a larger ω_n than the ω_n in the controller model but the relationship to the system model is not as clear, as will be demonstrated.

6.3.2.1 Modified LQG Controller Design

This section presents the results of the three MLQG design approaches as applied to the ideal mechanical-translational system from Section 6.1. The performance of these three designs will be analyzed in the follow section.

The first design is the MLQG with optimally selected filter parameter. The controller model matches the system model, and the ω_n for the filter model is optimally selected to minimize the position autocorrelation error. The controller gain matrix is determined with conventional linear quadratic regulator (LQR) techniques. The Kalman filter that completes the MLQG design is chosen using the optimization algorithm described in Section 3.2. The design algorithm implementation was kept simple. For this implementation, the optimization algorithm selects the Kalman filter from the possible filters corresponding to the designs based on discrete parameter points in the space. Finally, the standard MATLAB minimization algorithm *fminsearch* [31] is used to find the exact solution.

The results for the filter/controller selection process for the example problem are shown in Figure 6.4. Note that there is an apparent separation between the selected ω_n for the filter and for the designated controller. This separation indicates that the filter parameter, the undamped natural frequency ω_n, is greater than that of the system model. This corresponds to a filter model with faster dynamics, as predicted.

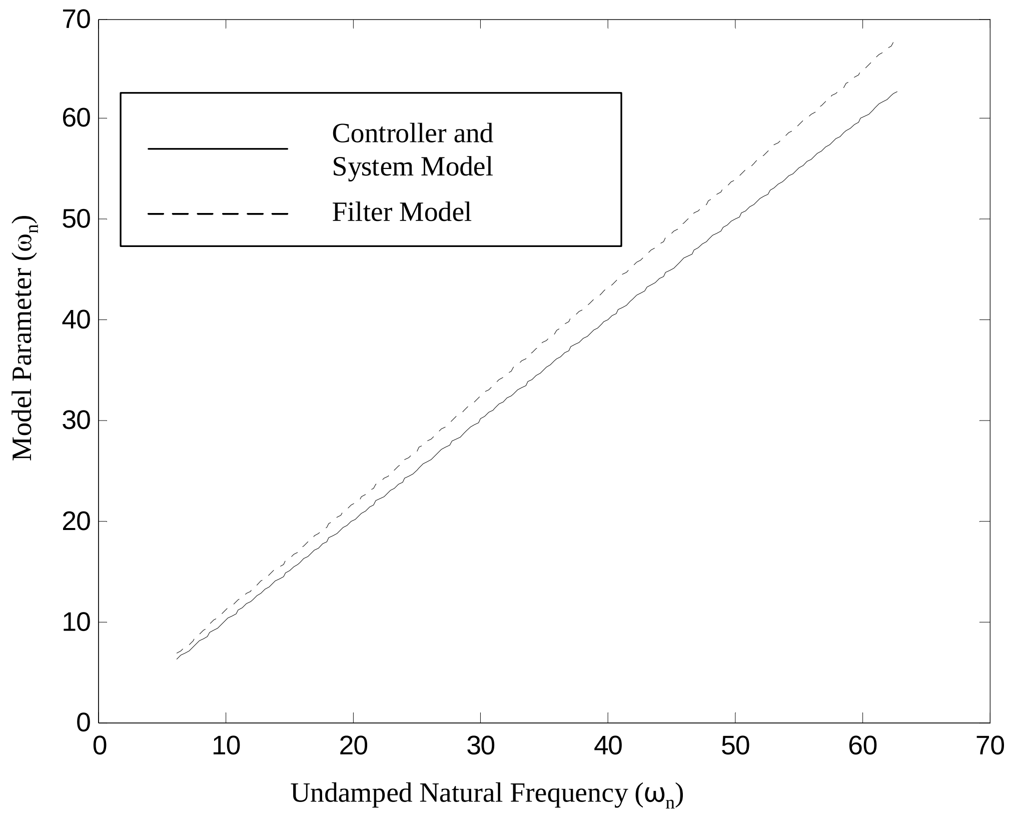

Figure 6.4 Filter and controller parameter selection for MLQG with optimally selected filter parameter

The second design is the MLQG with optimally selected controller parameter. In this case, the filter model matches the system model, and the ω_n for the controller model is selected optimally to minimize the position correlation error. The Kalman filter is designed using conventional techniques and the linear quadratic regulator that completes the MLQG design is chosen using the optimization algorithm described in Section 3.2. Similar to the previous case, the optimization algorithm selects the LQR controller from the possible controllers corresponding to the designs based on the discrete parameter points in the space. Again, a standard MATLAB minimization algorithm *fminsearch* [31] is used to find the exact solution.

The results for the filter/controller selection process for the example problem are shown in Figure 6.5. Again, note that there is an apparent separation between the selected ω_n for the filter and for the designated controller. As predicted, when the filter is fixed to the system model, the resultant controller has an undamped natural frequency that is less than that for the system and filter models. This smaller value for ω_n corresponds to a controller with a larger gain than a controller based on ω_n matched to the system and filter model. Similar to the previous design, the amount of separation increases as the assumed system undamped natural frequency increases. This separation for the two designs appears to be the same; though it is close, it is not exactly the same.

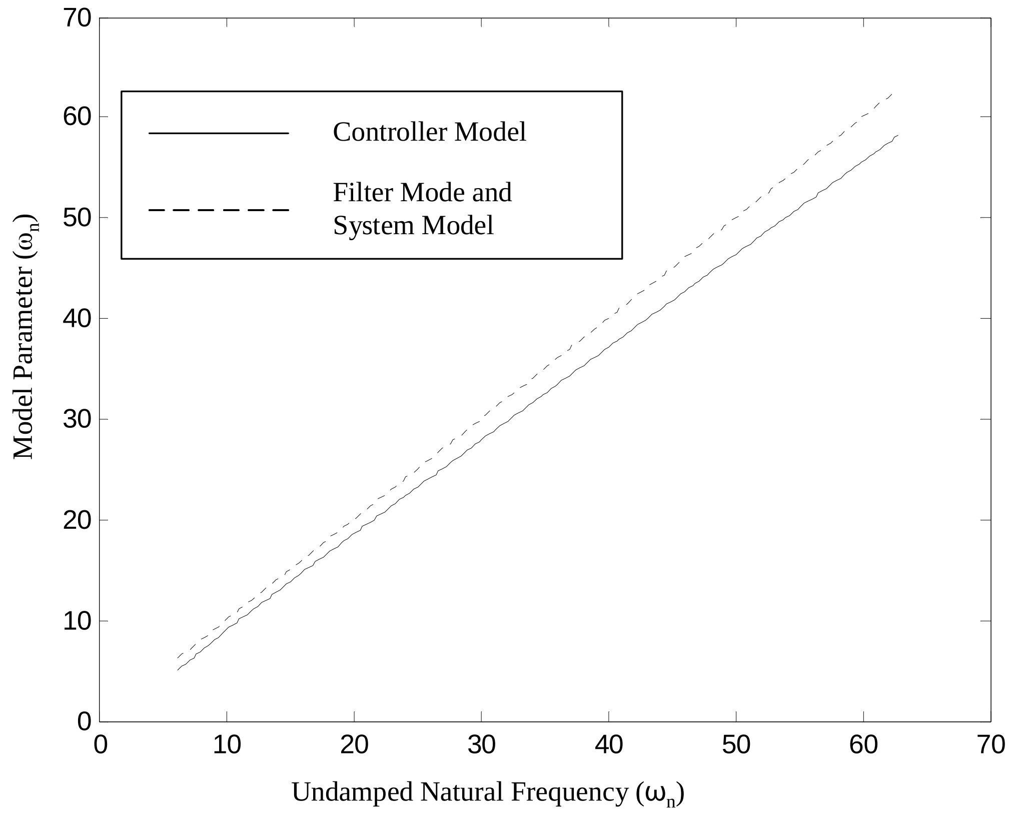

Figure 6.5 Filter and controller parameter selection for MLQG with optimally selected controller parameter

The final design is the generalized MLQG in which the natural frequency is selected optimally for the controller model and for the filter model. Each parameter value for the natural frequency may differ from the corresponding value for the system model. Similar to the previous two cases, the optimization algorithm selects the LQR controller and the Kalman filters from the possible designs based on the discrete parameter points in the space. Again, a standard minimization algorithm is used to find the exact solution.

Figure 6.6 shows the result for the filter/controller selection process for the example problem. Note that, as in the previous designs, there is a separation between the filter and the controller undamped natural frequency. Also consistent is that the degree of

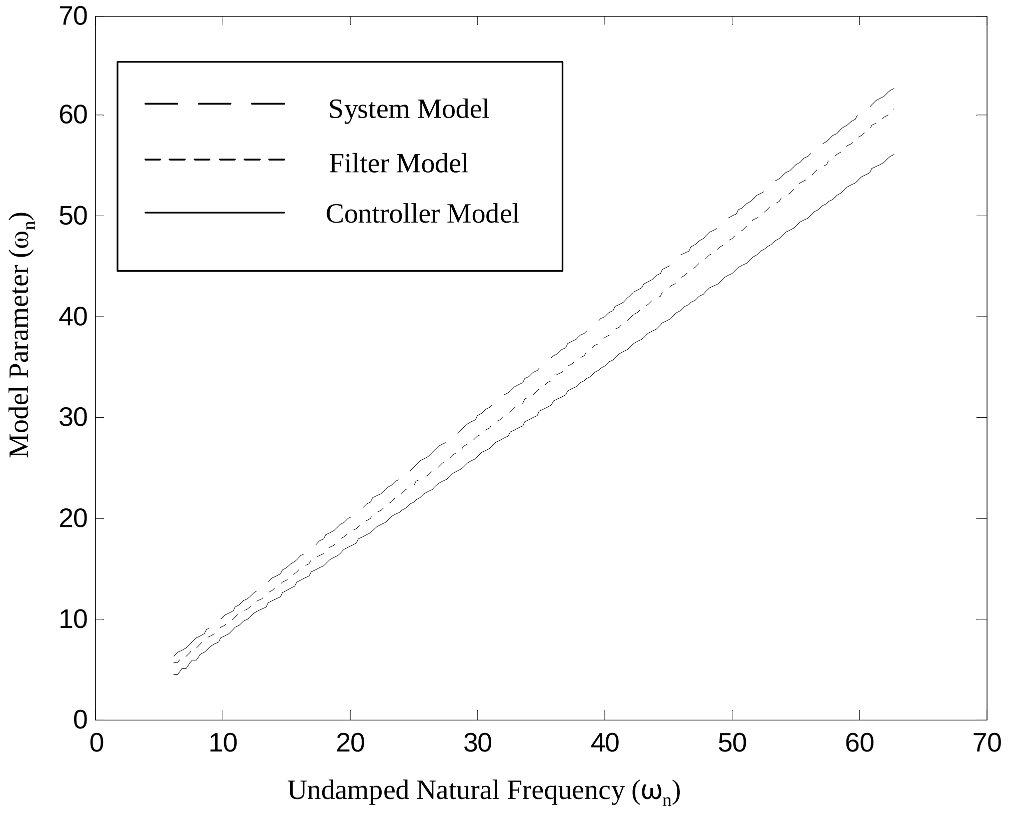

Figure 6.6 Filter and controller parameter selection for generalized MLQG

separation increases as the frequency increases. This degree of separation between the filter and controller looks the same as in the previous cases, though it is not precisely the same. However, what is obviously different for this design is that neither the filter model nor the controller model matches the system model. In fact, in terms of the previous discussions, the filter model for the design is slower than the system model. Clearly, the corresponding controller model has a gain that is significantly larger than either of the two previous cases in which the filter had a faster dynamic response than the actual system model. The larger gain is necessary to increase the dynamic response since the filter has slower dynamics.

6.3.2.2 Modified LQG Controller Evaluation

This section presents the results for the predictive analysis and Monte Carlo simulations for each of the three designs presented in the previous section. These results are compared to the baseline case from Section 6.3.1. Finally, the results from all three designs are compared against each other.

For the first design, the MLQG with optimally selected filter parameter, the results are shown in Figure 6.7 and Table 6.2. This MLQG outperforms the typical LQG design and this performance improvement becomes greater as the natural frequency increases. In fact, at the lower frequency, the performance difference is insignificant. The overall average performance across the entire parameter space has about a six percent decrease in the regulation error from that of the typical LQG design. The results also indicate that the predicted performance almost exactly matches the Monte Carlo simulations, to the point where the plots are almost indistinguishable and the predicted and computed performance averaged over the entire parameter space is almost the same.

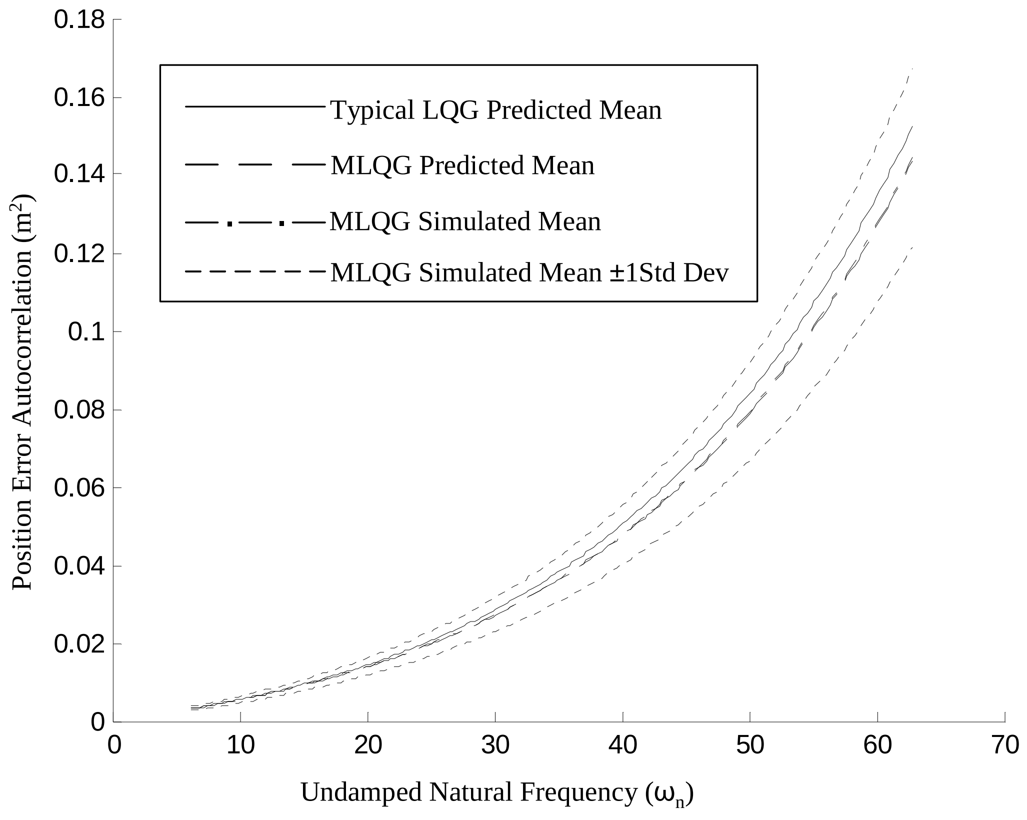

Figure 6.7 Performance of the modified LQG with optimally selected filter compared to typical LQG design

Controller	Predictive Measure Average Mean Squared Position (Meters^2)	Monte Carlo Simulation Average Mean Squared Position (Meters^2)
Baseline LQG	0.0503	0.0506
MLQG-Filter Selected	0.0475	0.0478
MLQG-Controller Selected	0.0445	0.0448
MLQG-Generalized	0.0439	0.0443

Table 6.2 Results for the modified LQG controller compared to the baseline LQG

The results for the MLQG with optimally selected controller parameter are presented in Figure 6.8 and Table 6.2. As in the previous case, this modified LQG design yields an improvement over the conventional LQG. The improvement becomes more significant as the natural frequency of the system increases, but again, the differences are negligible at the smaller parameter values. Here, the improvement at the higher frequencies is such that the performance of the typical LQG controller extends beyond the bounds of the mean ± one standard deviation region of the Monte Carlo analysis of the modified design. Overall, there is about a twelve percent improvement in the average regulation error across the parameter space in comparison to the typical LQG controller.

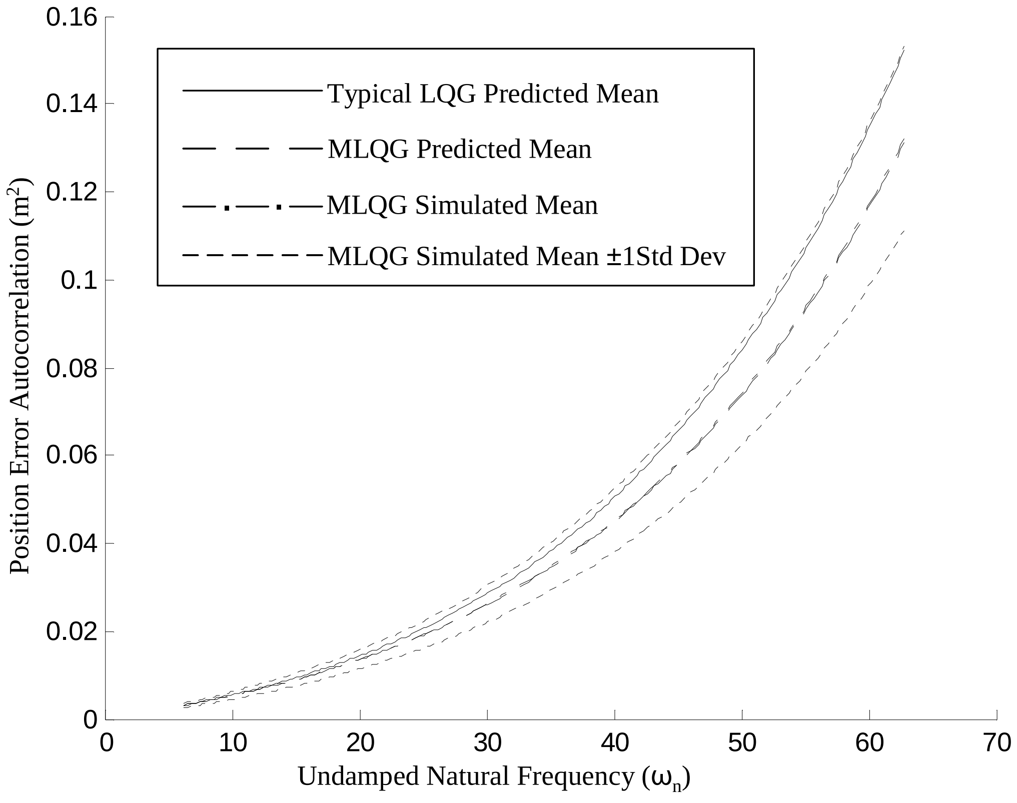

Figure 6.8 Performance of the modified LQG with optimally selected controller parameter compared to typical LQG design

Figure 6.8 shows considerably more improvement relative to the typical LQG results than does Figure 6.7. A comparison of the results in Table 6.2 with the previous MLQG-with-filter-selected approach demonstrates about a six percent improvement. Also, Figure 6.8 and Table 6.2 indicate that the Monte Carlo analysis matches very closely to the predictive results.

The results for the final design, the generalized modified LQG, are presented in Figure 6.9 and Table 6.2. As in the previous cases, the modified LQG design outperforms the typical LQG controller. In fact, these results look very similar to those in Figure 6.8 for the previous design. The performance of the typical LQG controller is close to the mean plus one standard deviation of the Monte Carlo analysis. There is

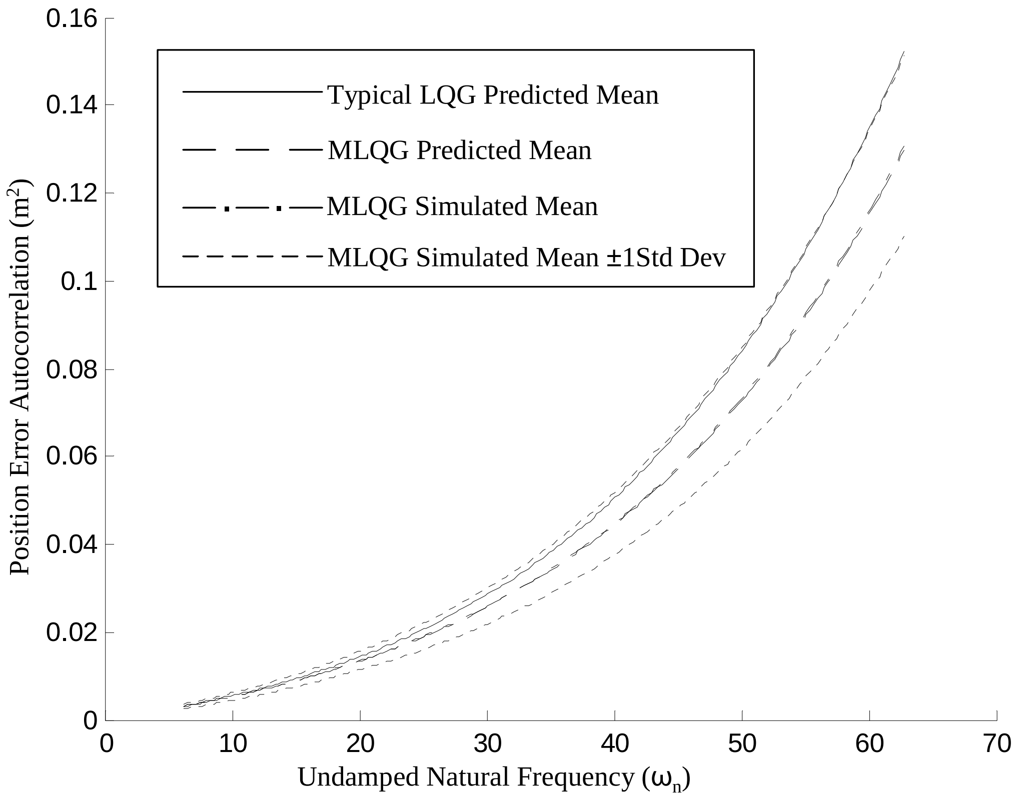

Figure 6.9 Comparison of the generalized MLQG results with the baseline LQG

about an eleven percent improvement over the typical LQG approach in the regulation error averaged across the parameter space. Also, Figure 6.9 and Table 6.2 indicate that the predictive analysis is almost coincident with the Monte Carlo simulation results.

A direct comparison of all three MLQG techniques along with the baseline LQG controller is presented in Table 6.2 and in Figure 6.10, which shows the predicted means from the previous figures for each LQG design method. From the graphical data, it is clear that all three modified designs outperformed the typical LGQ controller. The MLQG with an optimally selected filter yields the least amount of performance enhancement. In contrast, the other two modified designs are almost indistinguishable, with the generalized LQG design having a slightly greater performance improvement.

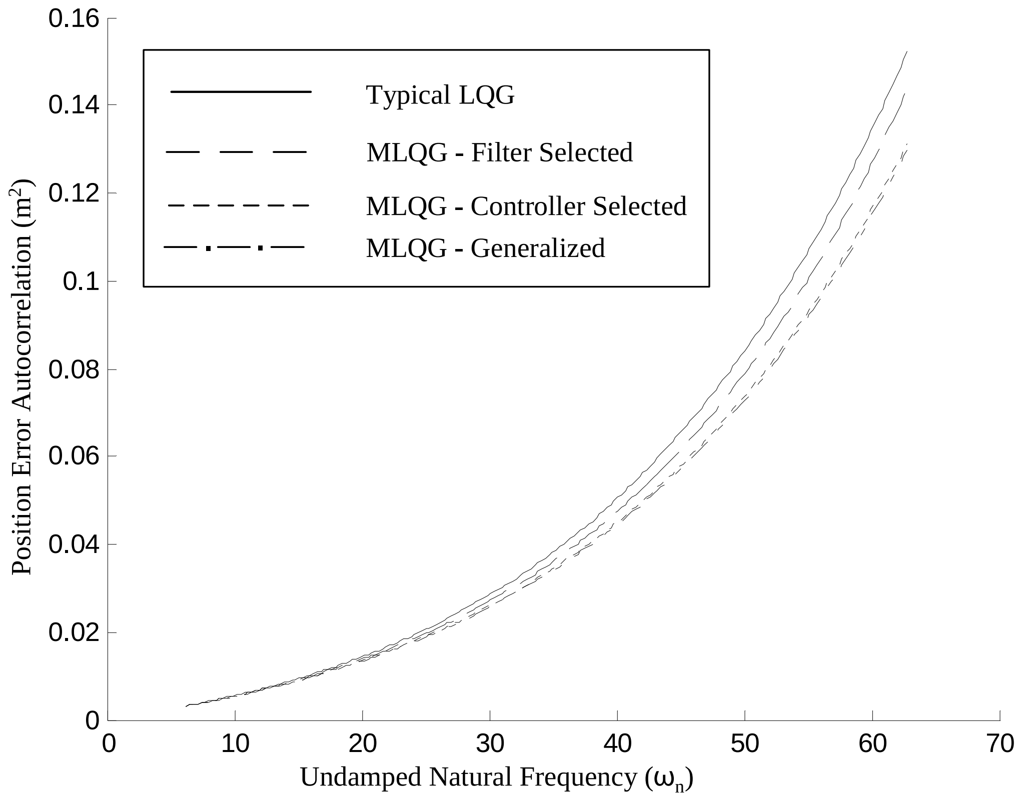

Figure 6.10 Comparison of baseline LQG and all three MLQG techniques

6.3.3 Modified LQG with Enhanced Robustness

This section discusses the implementation and evaluation of the MLQG with enhanced robustness as described in Section 3.3 for the ideal mechanical translational system described in Section 6.1. Up to this point, the evaluation of the MLQG controller designs has been for a single point that is the assumed true parameter value. The MLQG with enhanced robustness provides a design approach for the condition such that the value of the system parameter may deviate from the assumed true value. Each design approach will be assessed for its performance for a mismatch between the assumed true system operating point and a specified deviation from that point.

As in Chapter 3 and for the purposes of this discussion, robustness is defined as the performance adequacy over a region of parameter values, also referred to as a robustness ball. A center point in parameter space and a radius from that point describes the robustness ball. For the particular example in this chapter, the undamped natural frequency is the parameter space and it is one-dimensional. The center point is a specified value of the natural frequency and the radius will be the maximum that the natural frequency deviates from the center point.

As evaluated in the previous section, the steady state regulation error determines the performance. However, unlike the previous section, the regulation error is not measured at a single assumed value of the natural frequency parameter, but over the robustness ball. To evaluate the performance over the robustness ball, there are two measures that will be used. The first measure is the maximum regulation error over the entire region. The maximum regulation error gives a measure of the worst case performance when the true system deviates from the assumed value of the natural frequency. This maximum is determined by computing the performance at points sampled over the robustness ball. The second performance measure is the RMS of the output autocorrelation computed over the region. The sample points that are used to determine the maximum regulation error are used also to compute the RMS error. The RMS error in a sense gives the average performance over the entire region and is more representative of the anticipated performance of the controller over that region. As will be covered in the next subsection, the maximum and RMS regulation error are two measures of performance that are used in the design process.

6.3.3.1 Modified LQG with Enhanced Robustness Design

Whereas the previous modified designs assumed that an evaluation at a specified true value determines the performance, now the performance over an entire region will determine the optimal robust design. As previously discussed, the performance over the robustness ball is characterized by the computed RMS or the maximum regulation error. Either one of these measures can be minimized to determine the optimal controller design, but not both. The tradeoff between these two measures can be expressed in the related cost function given as:

$$J = \alpha J_{Max} + (1-\alpha) J_{RMS} \qquad (6.28)$$

where $0 \le \alpha \le 1$. Clearly, if α equals zero, the design algorithm tries to find the best RMS performance. On the other hand, if α equals one, then the worst case performance is minimized over the region. It would be up to the designer to determine the best α that would give the desired combination of average performance with protection against the maximum. Stability is not an issue since an unstable point in the robustness region will cause the evaluation of Equation (6.28) to become indeterminate and thus force the design algorithm or optimization method to move to a stable controller.

Implementation of the MLQG with enhanced robustness allows for two separate design variables, the size of the robustness ball and the value of α. For the robustness ball, a radius of 20 discrete sample points was selected arbitrarily. Since the number of samples in the region of the parameter space ranging from 2π to 20π has been specified arbitrarily at 200 points, the radius of robustness covers 20 percent of the admissible space. Each design will be evaluated for the 20-point-radius as well as a zero-point-radius ball. The latter case will allow a direct comparison to the MLQG designs from the previous section. It should be noted that for the 0 radius case, α does not affect the design since the region is only one point. For the 20-point-radius case, α equal to zero and one will be considered. These cases will demonstrate the tradeoff between design for best RMS performance and least maximum regulation error.

Chapter 3 discusses the MLQG with robustness that basically builds upon the generalized MLQG design. As an additional experiment, the robustness design also was applied to the modified LQG with selected controller or selected filter. Each of the designs was for the robustness ball of 20 points with α equal to 0 and 1. For the modified LQG with a selected filter, the design for both values of α and the 20-point robustness ball produced the same design as the previous non-robust design illustrated in Figure 6.4. It would seem logical that, since the region extends beyond the original design point, a filter with even faster dynamics would be desired. However, a faster filter will cause the original design at the center point (the original design point) to go unstable and thus not be adequate for the entire robustness ball. Recall that the controller model is fixed to the system model which is the robustness ball center. Likewise, for the MLQG with a selected controller, the design for both values of α and the 20-point robustness ball produced the same design as the previous non-robust design as shown in Figure 6.5. For this case, a larger gain is not required for the region that has a larger natural frequency than the center of the robustness ball. On the other hand, for the region that has a smaller natural frequency, a larger gain would perhaps improve the performance. However, a larger gain would cause the performance at the design point to go unstable. Recall that the filter model is fixed to be the same as the system model, which is the center point of the robustness ball.

The design for the generalized MLQG with enhanced robustness yielded significantly different designs for each of the design goals when compared with the generalized MLQG without robustness from the previous section. Of course the main difference for the generalized approach is that both the controller and filter can be placed differently from the assumed true value in parameter space. Thus, a faster filter may be selected since the controller gain is not necessarily designed based on the system model. Likewise with the filter model not fixed to the design point, a larger controller gain could be selected. One common characteristic of the generalized MLQG is that there is a separation between the filter and controller model parameters and that the filter model ω_n parameter is larger than the controller model parameter.

First, consider the case for which α equals one, with robustness radius of 20 samples. The resultant design is shown in Figure 6.11 plotted with the MLQG without robustness. Note the separation between the filter and the controller are similar, but not exactly the same for each design. A second observation is that the lines describing the

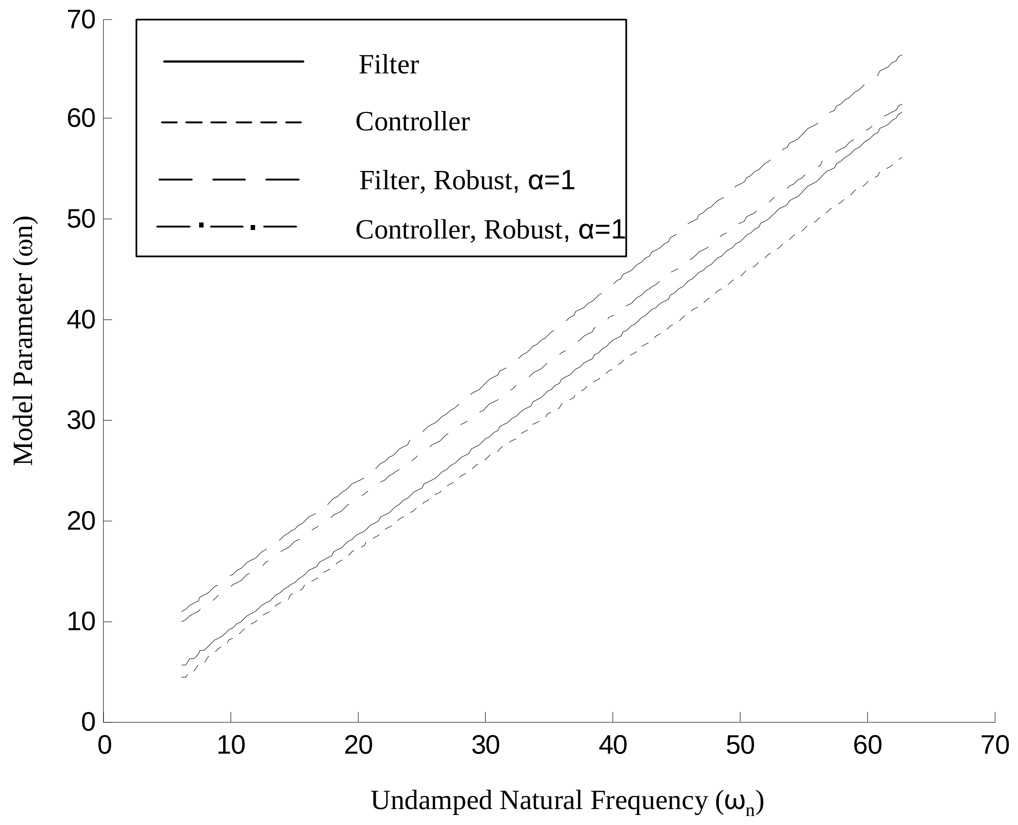

Figure 6.11 Filter and controller selections for the Generalized MLQG with and without robustness and α equal to one

filter design models are nearly parallel. Likewise is true also of the controller design models. Clearly, the design for robustness with best protection against the maximum regulation error over the region has selected faster filters than the nonrobust designs rather than larger controller gains. Unlike the MLQG with a selected filter, in which the controller model is fixed to the system model, the controller model is selected such that the gain decreases in accordance with a faster selected filter design model.

Next, consider the case for the robustness ball of 20 points with α equal to zero, as shown in Figure 6.12 plotted with the nonrobust generalized MLQG. Here again there is a separation between the filter and the controller similar to the design for α equal to one as well as a separation from the generalized MLQG without robustness. The main observation for this design is that the filter is once again faster for the robust design than for the nonrobust case. Likewise, the corresponding controller model has a smaller gain. However, what is different is that the separations between the robust and nonrobust designs are not as great as in Figure 6.11. This is not unexpected since the design objective is to obtain the best RMS performance over the region and so the filter does not have to be as fast as it would have to be to protect against the worst-case performance.

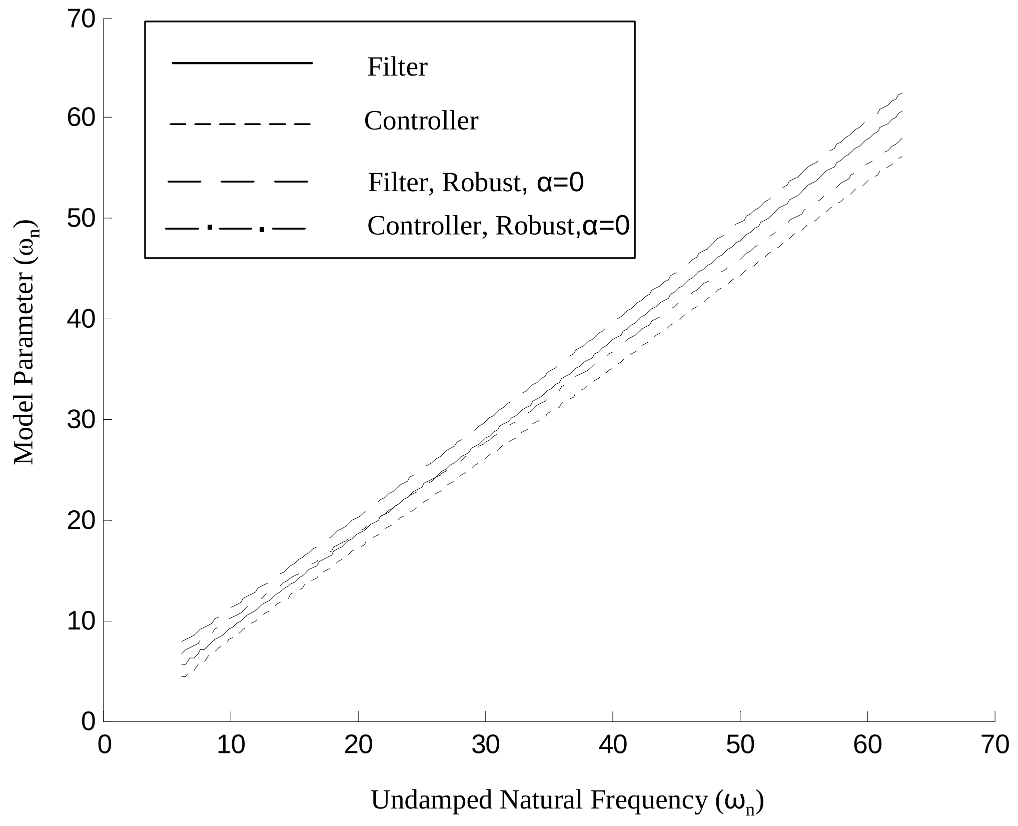

Figure 6.12 Filter and controller selections for the Generalized MLQG with and without robustness and α equal to 0

6.3.3.2 Modified LQG with Enhanced Robustness Results

Two different variables in the design specification are considered in evaluating the performance of the MLQG with enhanced robustness designs. The first design variable is the size of the robustness ball. The second variable is the type of performance, either the best RMS or smallest maximum regulation error, which will be considered for each specified robustness ball. The intent is to show how well the designs perform for their designated purposes as well as under other conditions. This analysis will demonstrate the interplay between the design specifications. The MLQG with enhanced robustness from the previous section will be evaluated. Also, the generalized MLQG without enhanced robustness will be used as the measure of any improvement.

First consider the evaluation of the controllers at a single point rather than over the entire the robustness region. For the MLQG with robustness, the assumed true value is the center point of the robustness region that was used as the basis for the design. Results for the predicted performance for the three different controller designs are shown in Figure 6.13. Clearly, the MLQG and the MLQG with enhanced robustness designed for the best average performance over the region have very similar performance results. The design for robustness did not require a sacrifice in performance at the assumed true value of the system. What is interesting is that the previous section demonstrates that these two designs are very different. In contrast, the MLQG with enhanced robustness designed for the least maximum position error over the region did not perform as well as the other two designs. This implies that the maximum position error at the center point of the robustness region, which is also the design point, is not representative of the maximum position error for the robustness region. Thus, if robustness is not needed and

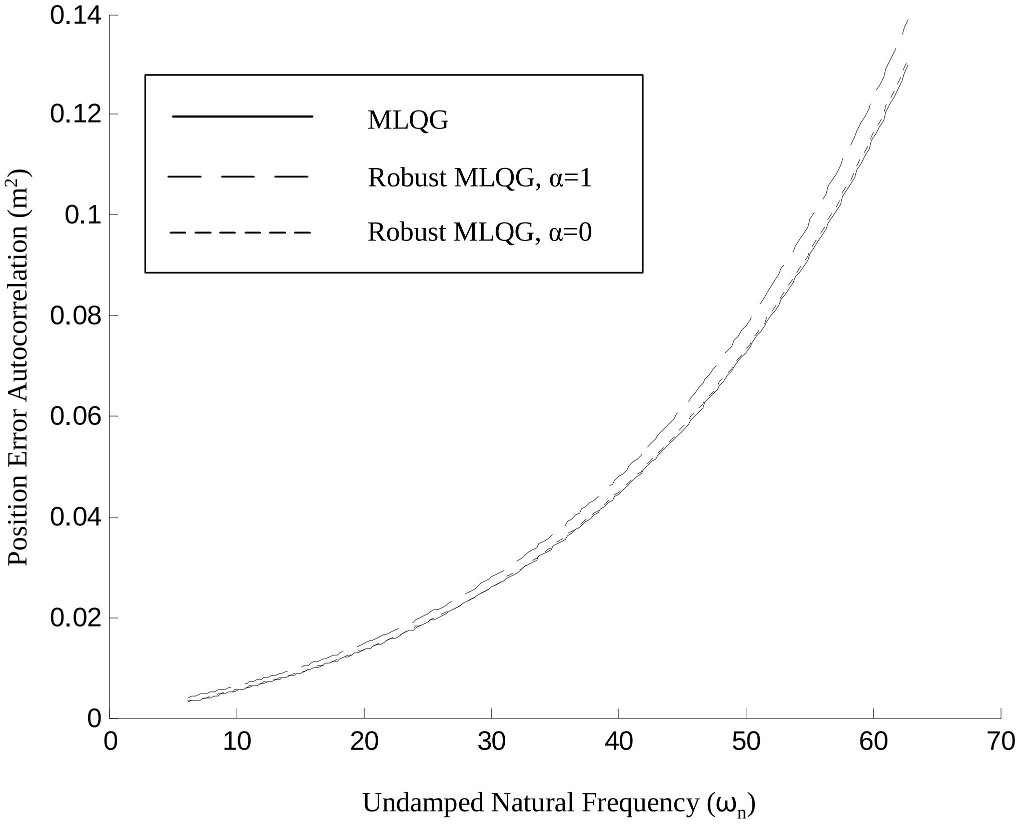

Figure 6.13 Comparison of the predicted performance for the generalized MLQG with and without enhanced robustness at only the design point

the assumed true value is close to the actual true value, then the design to protect against the maximum error would sacrifice performance.

Now it has to be determined how the designs perform for the designated robustness region of the parameter space. Consider the performance evaluation for the MLQ with enhanced robustness designed for the best maximum position error and the MLQG without robustness shown in Figure 6.14. Both the maximum and the RMS position error are evaluated over the parameter space for the two designs. As expected, the enhanced robustness design does better at protecting against the maximum position error than the MLQ without robustness. However, the MLQG without robustness

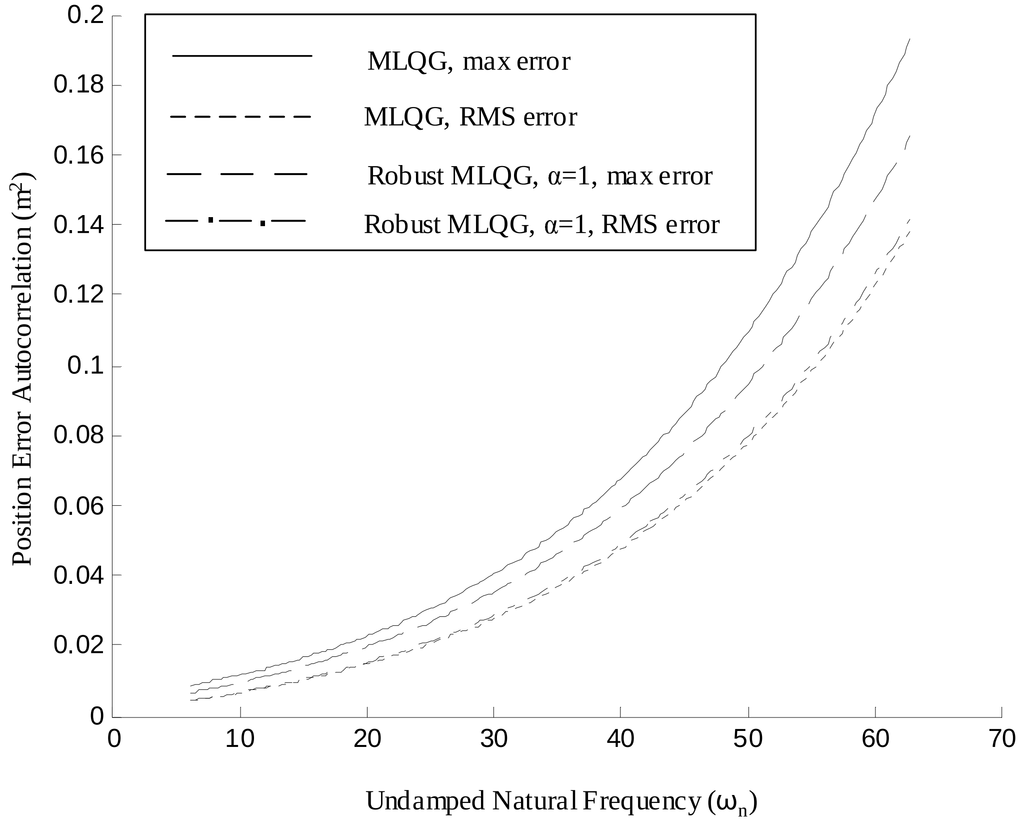

Figure 6.14 Comparison of the generalized MLQG with and without enhanced robustness (designed for best maximum position error) over the robustness ball.

enhancement performs the best for the RMS position error protection. This is only a slight performance improvement over the enhanced robustness design. From this empirical evaluation, the identifiable tradeoff is a slight sacrifice in RMS position error with the benefit of protection against the maximum position error.

Now consider the evaluation of the MLQG with enhanced robustness designed for best RMS performance as shown in Figure 6.15. Not surprisingly, the enhanced robustness design has the best performance for the RMS evaluation. However, the performance improvement is not significant. This is an indication that the performance at the center of the robustness region is representative of the RMS performance across the

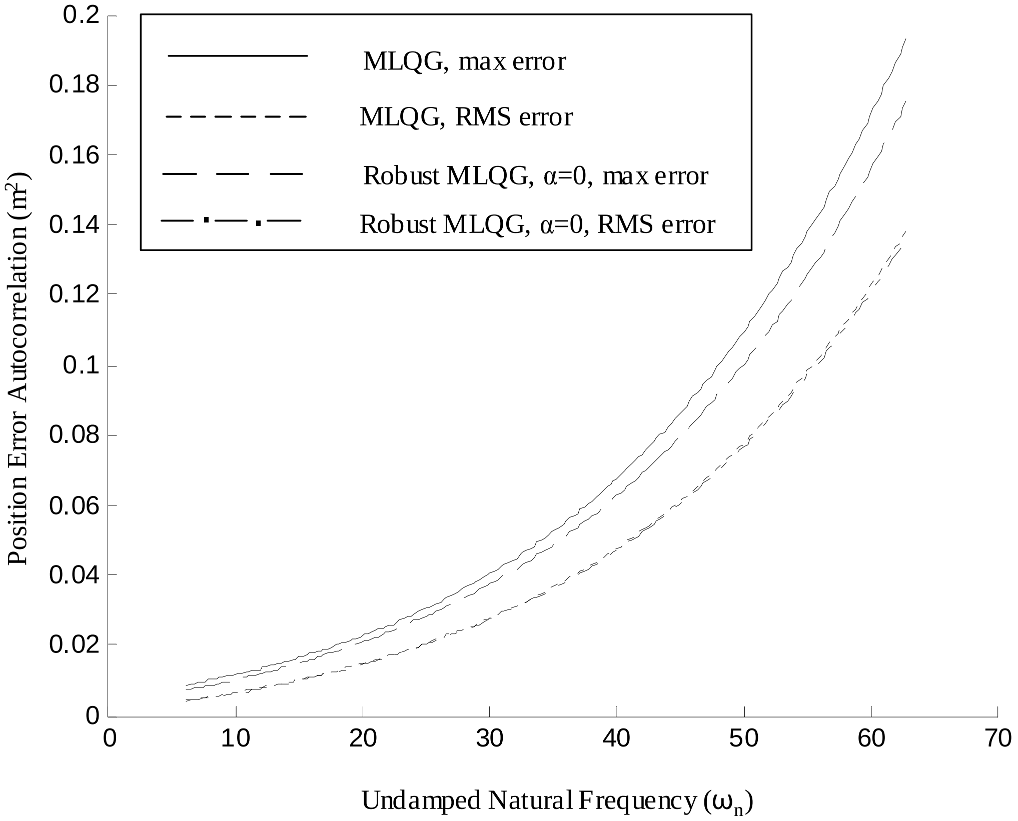

Figure 6.15 Comparison of the generalized MLQG with and without enhanced robustness (designed for best RMS position error) over the robustness ball.

entire region. Thus, there is only a slight improvement in the measure of the RMS position error performance. What is not expected is that there is a significant improvement in the maximum performance for the enhanced robustness design over the nonrobust design. Thus, in terms of performance, the tradeoff does not exist for the RMS error robustness design as it does for the maximum error robustness design. The RMS error robustness design outperforms the MLQG design.

Table 6.3 gives the average of the predicted position correlation error over the parameter space for the previously discussed controller evaluations. The typical LQG represented as the baseline case is included in this predictive analysis. It is of note that,

Controller	0 Pt Ball (Meters^2)	20 Pt Ball (Meters^2)	
		Ave	Max
Baseline LQG	0.0503	0.0529	0.0701
MLQG	0.0439	0.0472	0.0673
MLQG w/Robustness, 20 Pt, α=0	0.0443	0.0465	0.0617
MLQG w/Robustness, 20 Pt, α=1	0.0474	0.0485	0.0582

Table 6.3 Predictive analysis for the Modified LQG controller compared to the baseline LQG

in spite of the additional quality of robustness for all of the modified designs, they outperformed the baseline case. In terms of non-adaptive single controller design, this could impact future design approaches. The algorithms evaluated in this section allow the designer to specify the amount of robustness for given parameter(s) and can determine how the robustness affects the position correlation (mean square regulation error as an indicator of performance). Obviously, if robustness were not a requirement, the previous algorithm would be employed.

The entries in Table 6.3 clearly indicate that the controllers performed well for the conditions for which they were designed. In the graphical analysis it appeared that the MLQG with enhanced robustness designed for best RMS performance also performed well at protecting against the maximum position error. Though this controller outperformed the MLQ without robustness, it did not outperform the MLQG with enhanced robustness designed for least maximum position error. This analysis demonstrates that there is still a tradeoff between design variables in order to obtain the desired results. The techniques for enhanced robustness provide additional designs to consider. Table 6.4 shows the Monte Carlo analysis corresponding to the predictive analysis from Table 6.3. A comparison of the two tables indicates that the predictive

Controller	0 Pt Ball (Meters^2)	20 Pt Ball (Meters^2)	
		Ave	Max
Baseline LQG	0.0506	0.0523	0.0716
MLQG	0.0443	0.0466	0.0691
MLQG w/Robustness, 20 Pt, α=0	0.0447	0.0462	0.0629
MLQG w/Robustness, 20 Pt, α=1	0.0478	0.0485	0.0594

Table 6.4 Monte Carlo analysis for the Modified LQG controller compared to the baseline LQG

analysis is a good assessment of the simulated performance and can be used as an engineering design tool.

6.4 MMAC Evaluation

This section investigates the application of the MMAC techniques developed in Chapter 4 to the ideal mechanical-translational system described in Section 6.1. The first step for this investigation is to establish the baseline case for the MMAC performance by comparing the original optimal discretization algorithm by Sheldon [56] and the slightly modified version from Section 4.1. Next, the control implementation uses the Modified MMAC (M^{3}AC) approach proposed in Section 4.2. This approach is based on the Modified LQG from Section 3.2 and evaluated in the previous section. The natural extension to the M^{3}AC is the application of robustness enhancements to the Modified LQG from Section 3.3, which is evaluated next. The final MMAC design technique for evaluation is the Generalized MMAC discussed in Section 4.4. As part of the evaluations of MMAC techniques, the variance of the proximity measure as developed in Section 4.5 will be computed and analyzed.

6.4.1 Baseline MMAC

In this section, implementations of the MMAC for the ideal mechanical-translational problem compare the original optimal discretization algorithm by Sheldon [56] and the proposed modification from Section 4.1. The results from the analysis will serve as the baseline case for comparison of the subsequent evaluation of the MMAC techniques.

Recall that Sheldon's algorithm to discretize the parameter space basically uses a three-step process. First, filters are designed based upon models specified by arbitrary values of the design parameter, in this case ω_n. For the second step, the open loop MMAE equations are used to select the filter that is closest in probability to the assumed true system. Finally, the *closed loop* MMAC equations are used to evaluate the output position correlation for the assumed true system, given the selected elemental filter and controller. The process is repeated for discrete values across parameter space in order to compute the cost to be minimized. The modification uses the state estimates from the *closed loop* position correlation computation to determine the closest filter in probability to the true system. From an impact on design computation, the modification obviously cuts out a step. Since the MMAC is a feedback controller, the *closed loop* MMAC equations should provide an MMAC design and evaluation that more accurately reflects the architecture.

6.4.1.1 Baseline MMAC Design

The design for the MMAC structure places elemental controllers in parameter space according to the discretization scheme discussed in Section 4.1. As was specified in Sheldon's example, the MMAC has three elemental controllers to place in parameter space. The optimal placement minimizes position regulation error correlation over the parameter space given by the cost function developed in Chapter 4. For Sheldon's algorithm, the elemental controllers are placed in parameter space at:

$$A_S = [25.78 \quad 41.49 \quad 55.10] \tag{6.29}$$

For the modified discretization of Section 4.1, the elemental controllers are placed at:

$$A_M = [23.91 \quad 42.13 \quad 55.21] \tag{6.30}$$

For this implementation of the MMAC, the design parameter ω_n values specified by the members of the sets A_S and A_M above designate the design models for the filter and controller. Notice that there is only a slight difference for the second and third parameter models. This should not affect the performance at the larger values in the parameter space. However, the modified discretization decreased the value of the first filter's parameter. Normally in this application, underestimating the parameter ω_n within a single elemental controller will degrade the control performance dramatically as compared to overestimating it. Thus, in the case of the first filter, the improved performance must be greater than the sacrifice in performance elsewhere caused by moving the filter.

6.4.1.2 Baseline MMAC Results

Analysis not only compares the two design approaches, but also delineates the effectiveness of the prediction of performance. These results are compared to the LQG analysis of the previous section. Evaluation of the MMAC with the change to the discretization scheme will serve as the baseline case for comparison with proposed MMAC and MMAE-based controller design approaches.

Clearly, the modification to the original Sheldon algorithm only had a minor improvement on the performance when comparing the results of the performance for the two design approaches, as indicated in Table 6.5. The intent for the proposed

Controller	Predictive Measure Average Mean Squared Position (Meters^2)	Monte Carlo Simulation Average Mean Squared Position (Meters^2)
Baseline LQG	0.0503	0.0506
MMAC (Sheldon)	0.0540	0.0539
MMAC	0.0533	0.0538

Table 6.5 Results for the MMAC analysis

modification was not necessarily to show improved performance, but to demonstrate that the closed loop equations for the predictive analysis could be used rather than the open loop as proposed by Sheldon and determine if there is any advantage. A comparison of the predicted performances as shown in Figure 6.16 indicates that the main differences

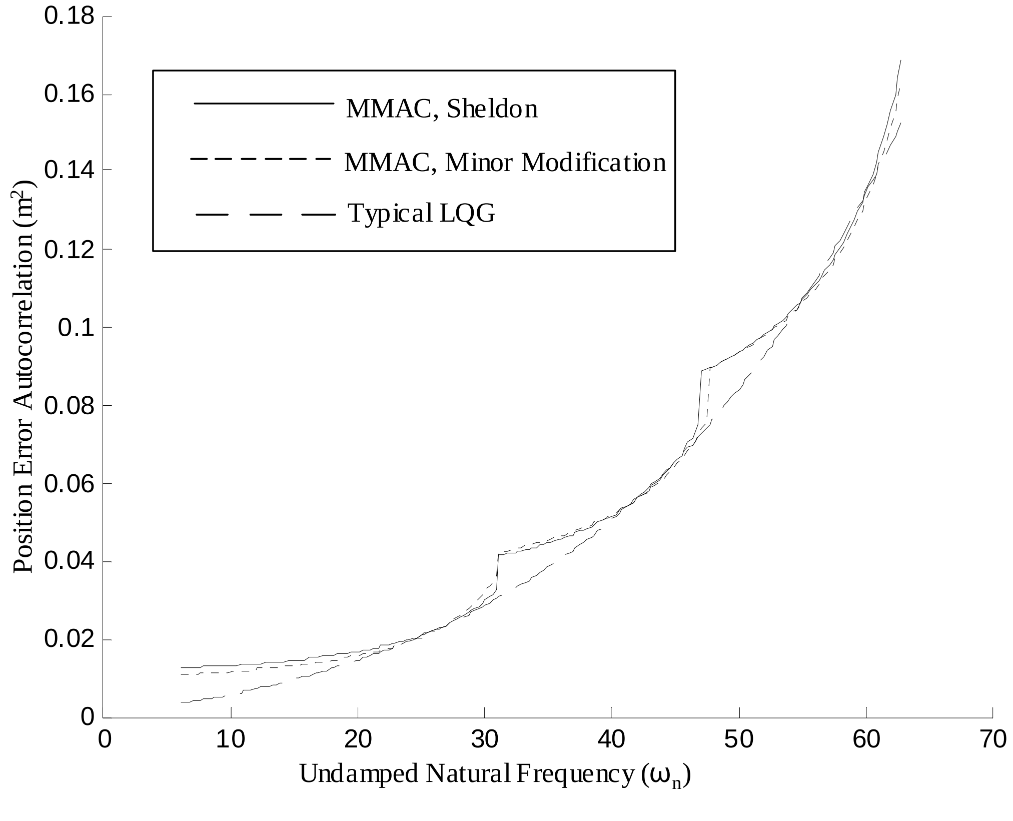

Figure 6.16 Predicted MMAC performance using the Sheldon algorithm and the modified approach overlaid with the baseline LQG controller performance

are at the transition points as well as the low and high end of the parameter space. Otherwise, the predicted performances are essentially the same. These points are where the filter and controller mismatches are the greatest. Finally, as expected, both designs are bounded below by the LQG evaluations at each point in parameter space. The LQG evaluations assume an elemental LQG controller at every point in parameter space, whereas the MMAC was limited to three.

Figure 6.17 shows the predicted performance of the modified MMAC overlaid with the Monte Carlo simulation results (mean and mean ± 1 standard deviation, plotted as a function of ω_n). The Monte Carlo simulation performance at both of the transition

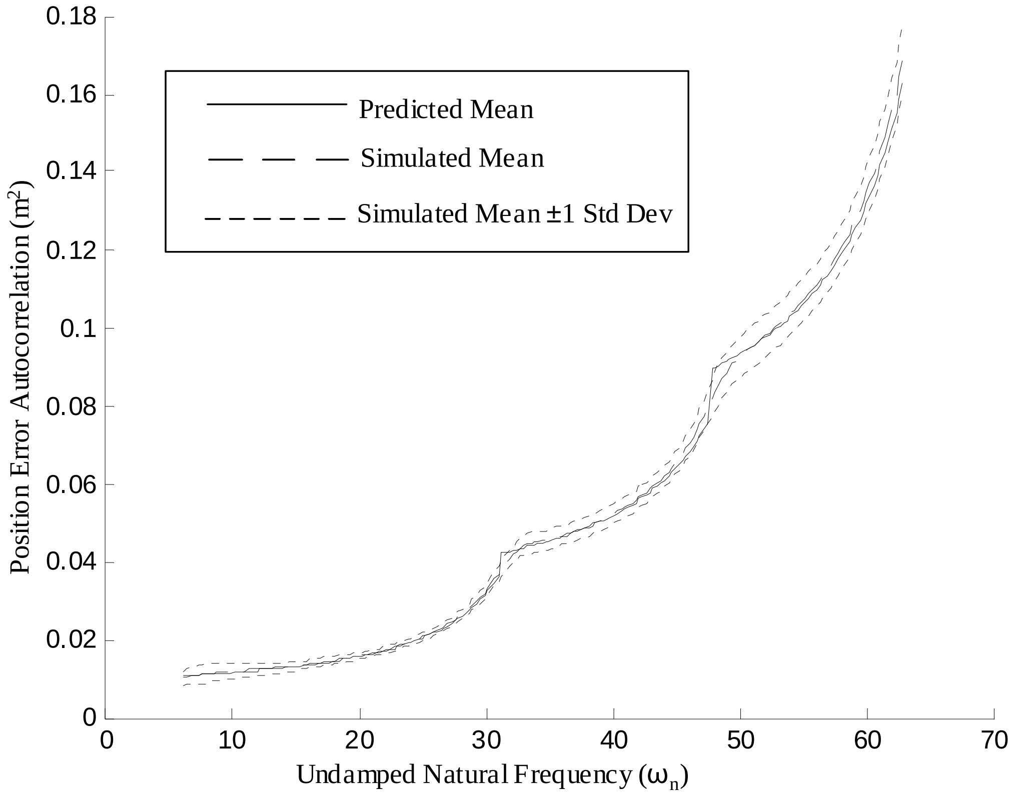

Figure 6.17 Monte Carlo simulation of the modified Sheldon designed MMAC overlaid with the predicted performance

points and the high end of the parameter space are the regions where the performance degrades the most. However, except at the transition points, the predicted mean stays within the bounds of the simulated mean ± one standard deviation. There, the prediction is slightly greater than the mean plus one standard deviation value. Of course, it is always better to have improved performance over the predicted. Overall, the predictive analysis using the MMAC with the modified discretization approach closely matched the simulated performance.

The fact that the predictive analysis for the MMAC with modification may perform well at the transition points, especially at the points in parameter space with larger gains, is particularly important to the follow-on techniques that rely on controller models that have greater associated gains. For this problem, the larger controller gains correspond to models that have smaller values of the paramter, ω_n. In order to make a fair comparison of the evaluations, a predictive performance evaluation using the closed loop equations was applied to the MMAC parameter locations found using the discretization with the open loop equations as specified in Equation (6.28). Figure 6.18 shows this predictive analysis as compared to the Sheldon predictive analysis, along with the simulation mean. This comparison shows that, at the lower transition point, the predicted mean using the closed loop performance equations does slightly better than the predicted mean as computed by Sheldon's original performance equations. For the remaining parameter space, the predicted mean is very similar for both performance evaluations. It is important for the discretization that the predictive analysis is accurate, especially when larger controller gains may be used, as is the case with models with smaller values of ω_n.

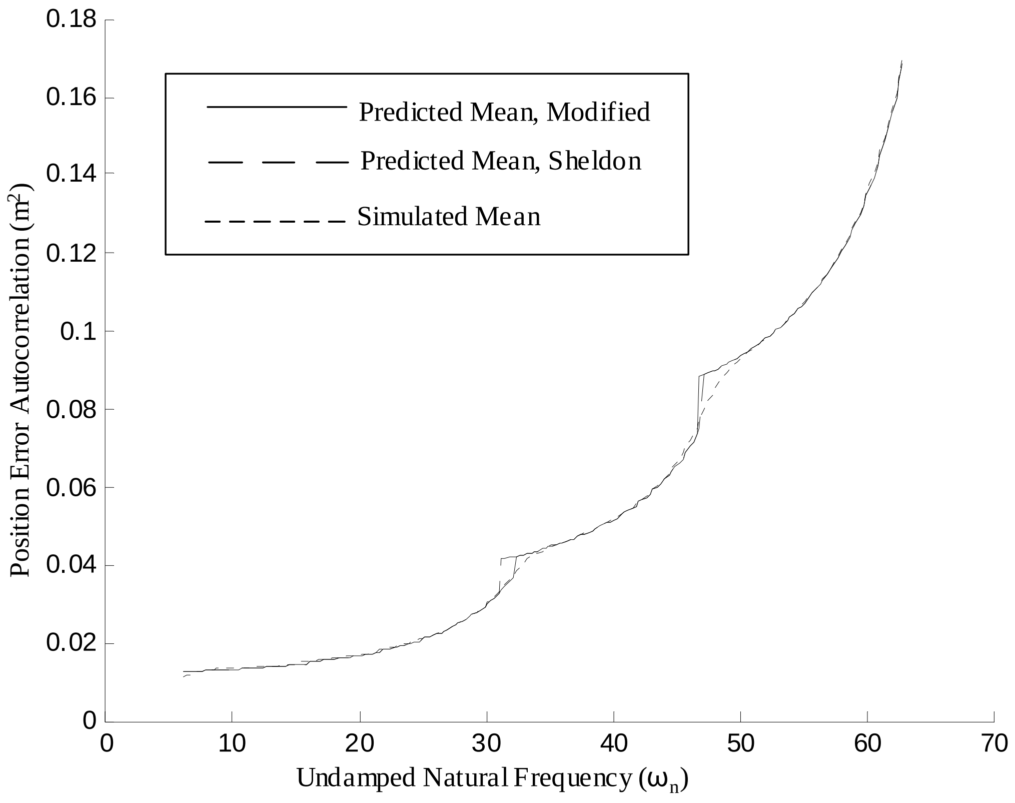

Figure 6.18 Predictive analysis and Monte Carlo Simulation for the MMAC with modification overlaid with the predicted mean of the original MMAC discretization

6.4.2 Modified MMAC

In this section, an implementation of the MMAC for the ideal mechanical-translational problem uses the M^3AC design proposed in Section 4.3. Similar to the MLQG approach, the filter and the controller, which form the elemental LQG controller in the conventional MMAC architecture, are based on different models. Of the three possible MLQG design approaches discussed in Section 4.3, the MLQG with a selected controller will be used to design the elemental controller. This design approach matches the physical architecture of the MMAC. As is the case for the typical MMAC, the filter is placed in parameter space and along with the associated controller and then the performance over the parameter space is evaluated. Thus, the filter is fixed and the controller must be designed based on the predetermined filter location and corresponding design. Previous analysis indicates that the MLQG with a selected controller outperforms the typical LQG controller, and it is predicted that the benefits of the MLQG will carry over to the M^3AC.

6.4.2.1 Modified MMAC Design

For the M^3AC, the design process uses the MLQG with a selected controller rather than the typical LQG controller. The design process places the filter in parameter space and then finds the corresponding controller using the same optimization as was used for the MLQG with a selected controller from Section 6.3.2.1. Next, the evaluation of the position error correlation uses the designed filter and controller. A standard MATLAB minimization algorithm *fminsearch* [31] was used to determine the optimal filter locations to achieve the minimum position correlation over the parameter space.

For the M^3AC discretization, the filters for the elemental controllers based on the parameter ω_n were placed at:

$$A_{M^3AC\text{-filter}} = [24.47 \quad 43.46 \quad 56.28] \tag{6.31}$$

The corresponding controllers are designed for the models based on parameter ω_n were placed in parameter space at:

$$A_{M^3AC\text{-controller}} = [22.70 \quad 40.37 \quad 52.18] \tag{6.32}$$

As expected, the optimization routine placement of the controllers was offset to lower parameter values than those for the corresponding filters. This is consistent with the MLQG design from Section 6.3. The placement of the filters is not exactly as was found for the MMAC with modified discretization. They are offset to larger values of ω_n, which indicates that these faster filters with the larger controller gains can provide better compensation for the lower frequency systems.

6.4.2.2 Modified MMAC Results

The intent of the analysis of the M^3AC results is not only to show improvements over the conventional MMAC from the last section, but also to demonstrate the relationship to the MLQG elemental controllers. Since the MLQG elemental controller demonstrates an improvement over the typical LQG designs, then it is predicted that the M^3AC will have similar improvements over the typical MMAC. Additionally, the MLQG controller evaluated at each point in parameter space should be the limiting case for the M^3AC. Obviously, the three elemental controllers in the M^3AC will not perform better than the MLQG evaluated at numerous discrete points across the parameter space. Finally, this analysis also will show the effectiveness of the prediction when compared to the Monte Carlo simulations.

The summary of the compared performances in Table 6.6 indicate about a ten percent improvement for the position correlation averaged over parameter space for the proposed M^3AC compared to the typical MMAC. Figure 6.19 clearly indicates that the most significant performance improvements are at the higher values of the parameter ω_n. Also, as predicted, the MLQG controllers evaluated over the parameter space serve as the

Controller	Predictive Measure Average Mean Squared Position (Meters^2)	Monte Carlo Simulation Average Mean Squared Position (Meters^2)
Baseline MMAC	0.0533	0.0538
Modified LQG	0.0445	0.0448
M^3AC	0.0472	0.0476

Table 6.6 Results for the M^3AC controller compared to the baseline LQG

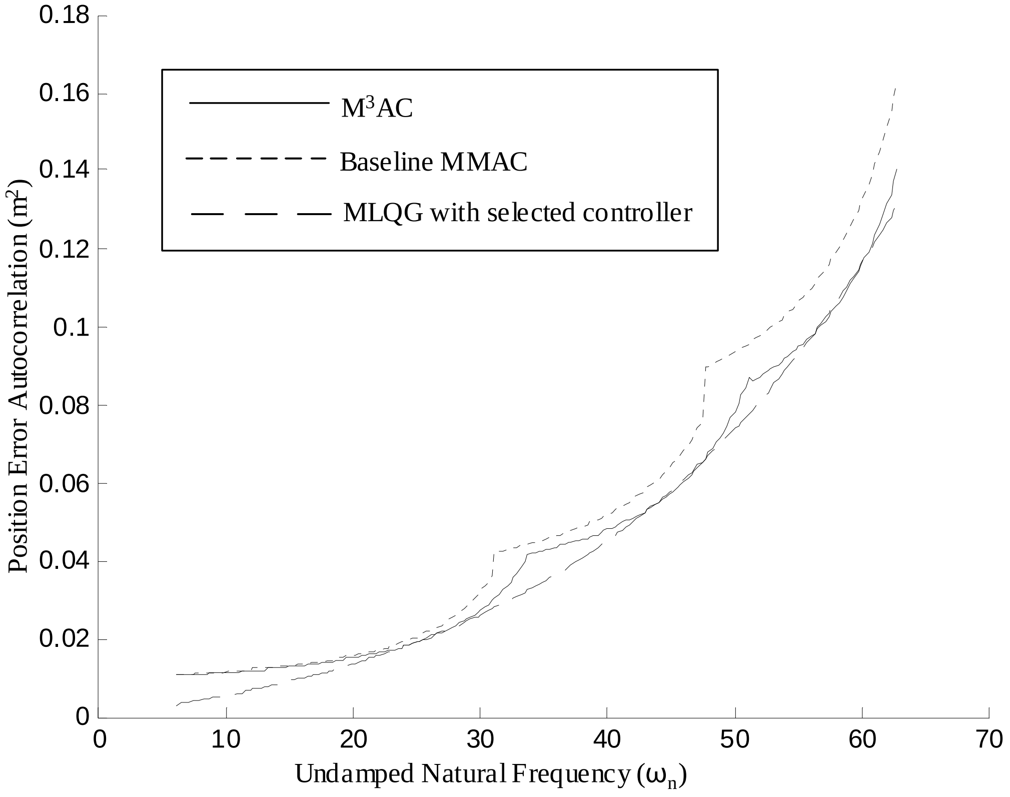

Figure 6.19 M3AC, Baseline MMAC, and MLQG controllers evaluated over the parameter space

lower bound on performance. However, the major differences are of course at the transition points, but also at the very smallest values of ω_n. As is the case with the typical MMAC, as the operating point is further from the elemental controller point, the performance degrades, with the peak degradation at the transition point.

Figure 6.20 demonstrates that the predictive analysis matches well with the Monte Carlo simulations, even at the transition points. Clearly, the predicted mean is completely contained by the simulated mean ± one standard deviation. Table 6.6 verifies the accuracy of the predictive analysis. The predictive mean averaged over the parameter

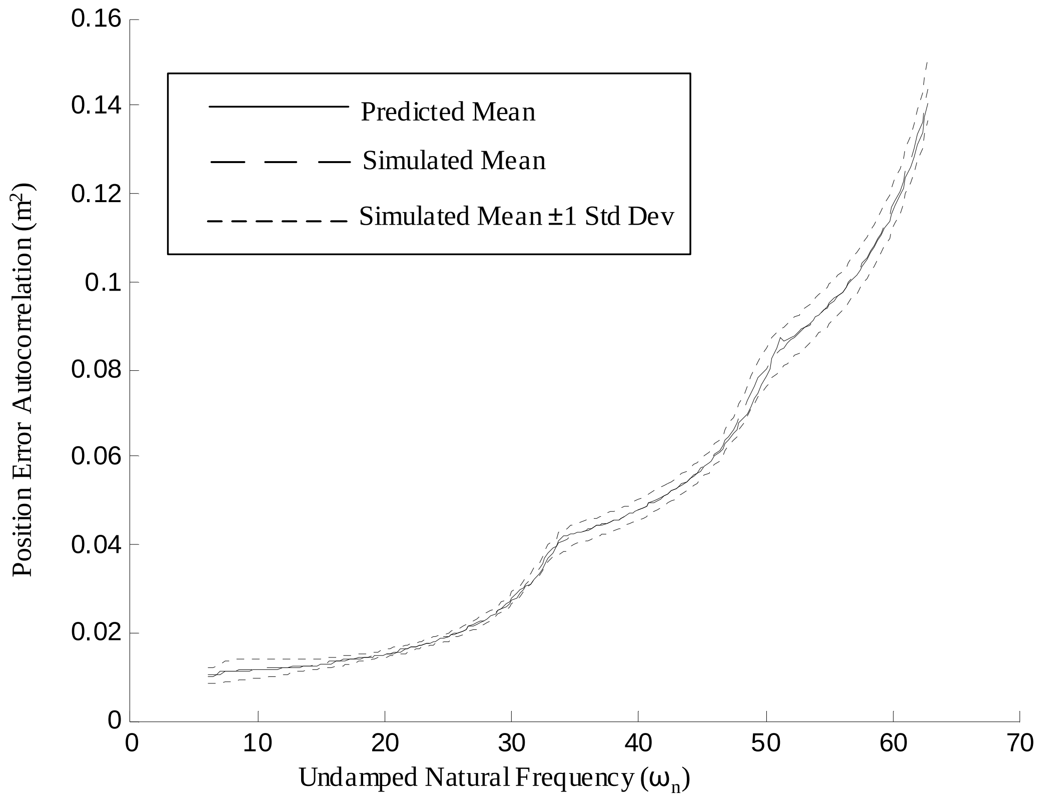

Figure 6.20 Monte Carlo simulation and predictive evaluation for the M^{3}AC

space is less than one percent different from the Monte Carlo simulation mean averaged over the same parameter space.

6.4.3 Modified MMAC with Enhanced Robustness

This section analyzes the implementation of the M^{3}AC with enhanced robustness design from Section 4.3 for the ideal mechanical-translational problem. As discussed in Section 4.3, enhanced robustness is an extension of the robustness techniques applied to the MLQG controller. The intent is to find the specific improvement of robustness around the region of the elemental controller location that also minimizes the position error over the entire parameter space. As reviewed in Chapter 4, at steady state a single filter is selected according to the proximity measure. The selected elemental controller must compensate for mismatches with, and for deviations of, the true system. Hence, the goal of the individual elemental controller design is to provide enhanced robust control when there are mismatches between the true system and the elemental controller.

As established in Chapter 4, the MMAC with enhanced robustness uses the MLQG with a selected controller gain as the elemental controller. The results from Section 6.3.3 demonstrate that the designs for the MLQG with a selected controller gain for enhanced robustness (for either the best RMS performance or the best maximum performance over the designated robustness ball) were not different from the non-enhanced robustness design. The robustness ball for those designs were 20 sample points or equivalently $\Delta\omega_n = 1.8\pi$. Therefore, it is most likely that the design for the M^3AC with enhanced robustness will be the same as the design of the M^3AC without enhanced robustness.

6.4.3.1 Modified MMAC with Enhanced Robustness Design

For the design of the MLQG with enhanced robustness, the engineer specifies the type of desired robustness and the design algorithm determines the filter placement, the controller design and the region of robustness. As computed for the MLQG with enhanced robustness, the cost function to achieve the desired type of robustness can be expressed as:

$$J = \alpha J_{Max} + (1-\alpha) J_{RMS} \tag{6.33}$$

This robustness design factor, α is determined by the design objectives and thus specified by the designer. However, for the robustness ball, the discretization process will determine its size. The region of robustness is dependent on the relative placement of the elemental controllers. The transition point between the elemental controllers is a gauge for the size of the robustness region. Of course, the locations of the transition points are determined by the elemental controller design and discretization.

The optimal design of the M^3AC with enhanced robustness followed the design process outlined in Chapter 4. For this implementation, three filters were designed for three different points in the parameter space. For each filter, the corresponding controllers were designed according to the design process for the MLQG with enhanced robustness given a specified robustness ball. Recall that the design of the MLQG involves a minimization of Equation (6.33) (using MATLAB's *fminsearch* routine [31]) across the region of robustness. Finally, given the three resultant elemental MLQG controllers with enhanced robustness, the average of the position error is computed for the parameter space. A MATLAB minimization routine was employed to minimize this cost. The resultant optimization returns the filter models, controller models, and the radius of robustness.

For the design of the MMAC, two values of α were considered; zero and one. For the case where α equals zero, the resultant design was:

$$A_{M^3AC\text{-robustness, filter}} = [22.06 \quad 41.23 \quad 55.86] \tag{6.34}$$

$$A_{M^3AC\text{-robustness, controller}} = [20.46 \quad 38.28 \quad 51.80] \tag{6.35}$$

$$\text{Radius of Robustness} = [10.12 \quad 9.80 \quad 7.19] \tag{6.36}$$

For the case where α equals one, the resultant design was:

$$A_{M^3AC\text{-robustness, filter}} = [22.11 \quad 41.25 \quad 55.85] \tag{6.37}$$

$$A_{M^3AC\text{-robustness, controller}} = [20.50 \quad 38.30 \quad 51.79] \tag{6.38}$$

$$\text{Radius of Robustness} = [3.47 \quad 9.11 \quad 3.60] \tag{6.39}$$

The two different designs are nearly the same as far as the filter and controller parameter locations are concerned, but not the radii of robustness. Similar filter and controller model locations for the two different designs should result in similar control performances. However, the significant difference in the size of the robustness regions will not affect the resulting performance. The radii of robustness for the *lower* and *upper* filters given in Equations (6.36) and (6.39) are greater for the RMS position error than that for the maximum position error. Since the system is second order, the resultant control function will be parabolic. Thus, the difference between the robustness radii is due to the filter model not being located at the minimum of the corresponding performance curve.

6.4.3.2 Modified MMAC with Enhanced Robustness Results

The results of testing the Modified MMAC with enhanced robustness are summarized in Table 6.7. Consistent with the performance evaluation of the previous experiments, the average mean squared and average maximum squared position error are computed over the entire range of the parameter space. Neither the design for minimizing the RMS error

Controller	Predictive Measure Average Mean Squared Position (Meters^2)	Monte Carlo Simulation Average Mean Squared Position (Meters^2)	Monte Carlo Simulation Average Maximum Squared Position (Meters^2)
Modified LQG	0.0445	0.0448	N/A
Baseline MMAC	0.0533	0.0538	0.0594
M^{3}AC	0.0472	0.0476	0.0524
M^{3}AC w/Robustness, α=1	0.0471	0.0476	0.0522
M^{3}AC w/Robustness, α=0	.00471	0.0476	0.0522

Table 6.7 Results for the M^{3}AC with robustness compared to the baseline LQG

nor the design for minimizing the maximum error significantly improved performance over the M^{3}AC design. It did not seem to matter that the filters and controllers for both designs are significantly different from those of the M^{3}AC specified in Equations (6.31) and (6.32). The parameter for both the filters and controller gain models are shifted to lower frequencies for all three elemental controllers.

The reason for the difference in design as compared to the M^{3}AC is how the optimal controller gain is found. The M^{3}AC only finds the controller gain that corresponds to the filter location as the assumed true value. These elemental controllers (consisting of the filters and the selected controller gains) are then evaluated across the range of parameters. Now the controller gain selected for the enhanced robustness for either RMS or maximum position error is for the range specified by the radius. Hence, a different controller was selected than was for the M^{3}AC which will in turn affect the filter placement.

The robustness factor α did not affect the performance results as seen in Table 6.7. However, the radii of robustness for the filters for the two designs were not the same. The overriding factor in the design is that the filter placement has to minimize the average position correlation over the parameter space. The design optimization found the corresponding controller gain that minimizes the RMS or maximum position error over the robustness radius. For this experiment, the type robustness over the individual regions surrounding the elemental controllers does not matter.

6.4.4 Generalized MMAC

This section analyzes the implementation of the generalized MMAC (GMMAC) for the ideal mechanical-translational problem established in the first section. For the GMMAC architecture, the design adapts the design strategy for the generalized Modified LQG (MLQG) controller. Recall from Chapter 3 that, for the generalized MLQG, the filter and controller models both may be different from the assumed truth model. Thus, to implement the generalized MLQG in the typical MMAC architecture, the controller gain is replaced with an MLQG controller gain. Rather than state estimate information being fed to the elemental controller gain, measurement information is fed through to the MLQG controller elements to compute the control. In the generalized MMAC, the filters in the MMAE bank provide the residual information for the computation of the probability that will be used to weight the elemental MLQG controller output.

Recall that from the previous evaluations of the MLQG with selected controller and the generalized MLQG, the performance improvement over the parameter space was not that significant. However, the filters and controllers for each design approach were significantly different. Thus, for the two-state problem, there was only marginal benefit of the generalized MLQG controller. It is assumed for the MMAC application of the generalized MLQG controller, any improvement will be similar to the MLQG enhancement and will not be significant.

6.4.4.1 Generalized MMAC Design

The design of the GMMAC required two sets of optimizations. The first is for the filter locations for the MMAE filters that provide the residual information and the second is for the elemental MLQG controllers. The overall design begins with specifying the values of the parameter ω_n for each filter in the MMAE portion. For each filter, an MLQG is designed using the optimization procedures from Section 6.3. The next step is to compute the average position correlation evaluated across the parameter space. A

MATLAB routine was used to determine the filter locations and the corresponding MLQG that minimizes the average position correlation. The optimization resulted in the following parameter specification for the filter models, and the generalized MLQG filter and controller models:

$$A_{filter} = [24.39 \quad 43.61 \quad 58.08] \tag{6.40}$$

$$A_{MLQG\text{-}filter} = [22.66 \quad 41.39 \quad 55.90] \tag{6.41}$$

$$A_{MLQG\text{-}controller} = [21.01 \quad 38.42 \quad 51.83] \tag{6.42}$$

As was demonstrated for the MLQG from Section 6.3, the values for the parameters of the controller gain models are less than the corresponding filter models. Both filter and controller ω_n values for the MLQG are less than the ω_n of the assumed system model. The assumed system models in this case dictate the filter locations of the MMAE portion. In fact, the MLQG results were used as the initial *guess* in the optimization. The minimization merely refined the generalized MLQG controller results to match the filter locations more precisely.

6.4.4.2 Generalized MMAC Results

Since the GMMAC is based on the work from the generalized MLQG controller, it is expected that the results for the generalized MLQG evaluated over the parameter space will serve as the lower bound on performance. As shown in Figure 6.21, the predicted performance of the GMMAC does not outperform the predicted MLQG at any point in parameter space. As is typical of the MMAC, the greatest difference in performance occurs at the points at which the probability flows from one filter to the next and at the ends of the parameter space. These of course are the regions of the greatest parameter mismatch between the filter locations and the system. Recall that the evaluation of the

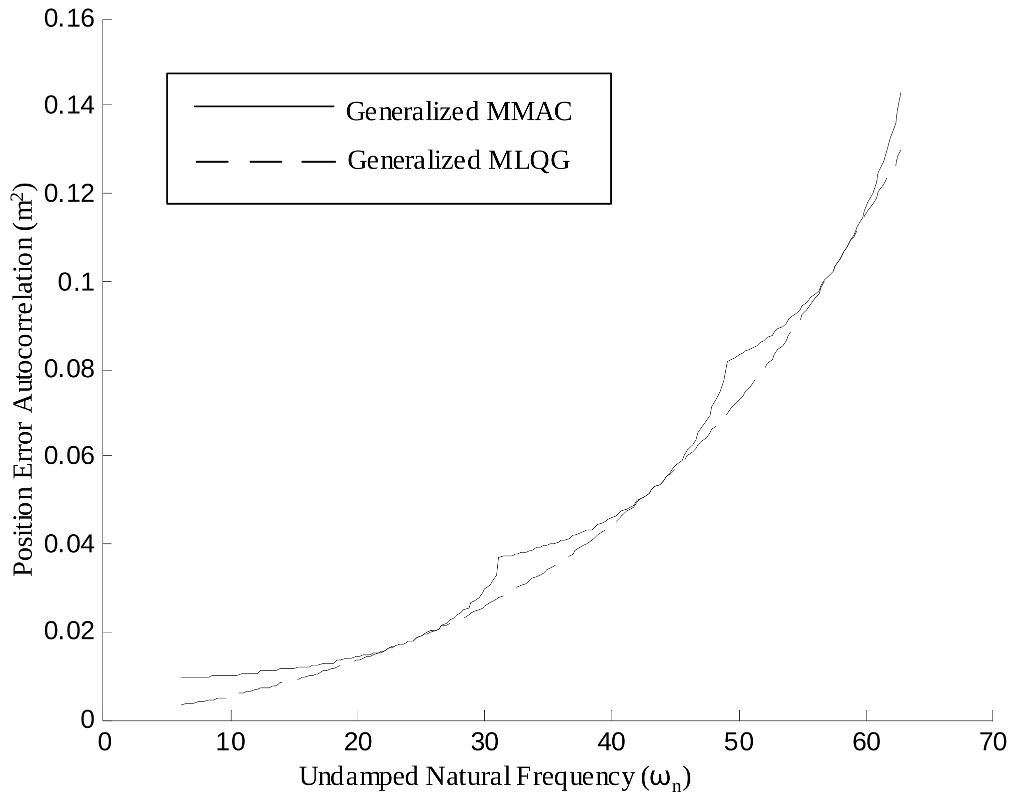

Figure 6.21 Predicted performance of the GMMAC overlaid generalized MLQG performance

MLQG assumes an elemental controller at every point in parameter space, which is impractical to implement. Table 6.8 shows that the difference in performance between the M^3AC implementation and the MLQG evaluations is about 7 percent. Hence, there is a relatively small sacrifice in performance for the practical implementation, which used only three elemental controllers.

Figure 6.22 shows the overlay of the Monte Carlo analysis and the predictive analysis of the GMMAC. The Monte Carlo analysis follows the predicted performance very closely even at the transition points as well as the upper and lower bounds of the

Controller	Predictive Measure Average Mean Squared Position (Meters^2)	Monte Carlo Simulation Average Mean Squared Position (Meters^2)
Baseline MMAC	0.0533	0.0538
MLQG-generalized	0.0439	0.0443
M^{3}AC	0.0472	0.0476
GMMAC	0.0472	0.0477

Table 6.8 Results for the GMMAC compared to the baseline MMAC

parameter space. The predicted analysis clearly falls within the bounds of the mean ± one standard deviation.

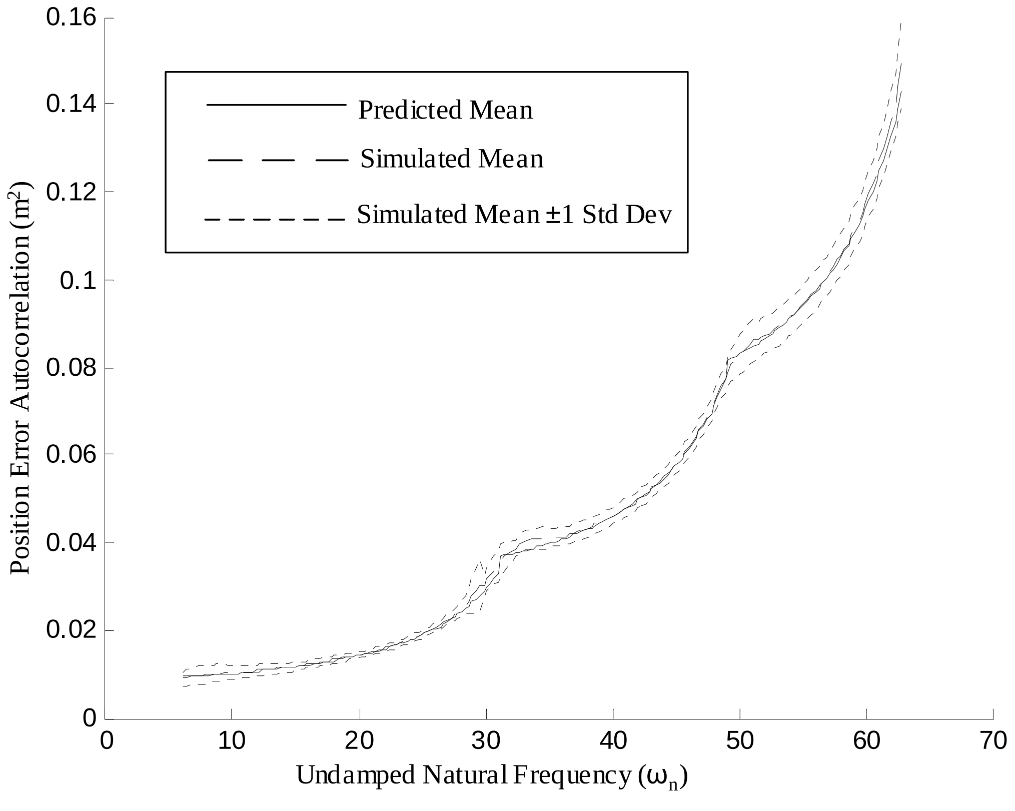

Figure 6.22 Predicted and Monte Carlo simulated mean of GMMAC

Though the generalized MMAC outperforms the typical MMAC, as indicated in Table 6.8 it produces almost the exact same results as the M^3AC. Thus, for this example, the additional overhead of maintaining three extra filters did not achieve any performance benefits. This is not unexpected since the MLQG with selected filters and controllers did not outperform the MLQG with a selected controller. Though it is beyond the scope of this current research to test this architecture fully, it would be beneficial to test higher order MLQG controllers as the control elements with reduced order filters that provide the control weighting information. This scenario would save much computation for implementation of large scale models.

6.4.4.3 Generalized MMAC with Optimized Parameter Estimation

This section investigates the design and performance of the GMMAC when the PFB has been optimized for parameter estimation rather than for control. From the previous evaluation, the GMMAC does not present a substantial advantage for control performance as compared to the M^3AC. However, as presented in Chapter 4, the advantage of the GMMAC is that the discretization of the PFB portion can be optimized for parameter estimation and the component MLQG controllers can be optimized for control for their regions of the parameter space (as determined by the PFB discretization). The control performance is expected to be superior to that of the typical MMAC but not the GMMAC optimized for control.

For this implementation, the design algorithm is very similar to that for the GMMAC in which the filter locations are optimized for control performance. The first step selects the PFB filter locations and then designs the optimal MLQG controller corresponding to each filter location and the associated range of the parameter values spaced for that filter. For the second step, rather than determining the position error, the parameter estimation error is integrated over the parameter space. The MATLAB optimization algorithm *fminsearch* [31] is then used to minimize this parameter estimation cost.

In order to evaluate the GMMAC optimized for parameter estimation, the results will be compared to those for the MMAE, the MMAC, the MMAC optimized for parameter estimation and the GMMAC optimized for control. The design for the typical MMAE, which for this experiment is optimized for parameter estimation, followed Sheldon's approach [56] and yielded filter locations:

$$A_{\text{MMAE-filter}} = [18.12 \quad 34.33 \quad 52.29] \tag{6.43}$$

The filter locations for the MMAC are the same as those found in Section 6.4 and repeated here for reference:

$$A_{\text{MMAC-Control}} = [23.91 \quad 42.13 \quad 55.21] \tag{6.44}$$

Clearly, the filter locations for the MMAC are different from those for the MMAE. For discretization for parameter estimation, the filters are spread across a greater range of the parameter space. The MMAE requires coverage of the parameter space to generate the estimates at the lower and upper ranges. The MMAC is not typically evaluated for parameter estimation performance and the lack of coverage of the parameter space will clearly degrade estimation performance. In this application, underestimation of the parameter will severely degrade the control performance versus overestimation of the parameter. Hence, the "low end" value of Equation (6.44) is much higher than the corresponding value in Equation (6.43).

Normally, the MMAC is not designed for optimal parameter estimation. However, it is a simple matter to replace the control performance measure with a parameter estimation measure for the optimization. The parameter estimation measure uses the position autocorrelation equations which are based on the modification to Sheldon's MMAC discretization approach discussed in Chapter 4. This optimization approach yields filter locations:

$$A_{\text{MMAC-Param}} = [17.57 \quad 34.92 \quad 52.88] \tag{6.45}$$

It is not surprising that Equation (6.43) is very similar to Equation (6.45), since the nature of parameter estimation requires a greater coverage over the range of the parameter space (and can afford to use lower discrete parameter values than an algorithm optimized for control could). It is expected that the parameter estimation performance should be very similar to that of the MMAE.

For comparison to the typical MMAC, the design results for the GMMAC optimized for control found in Section 6.4.4.1 are restated:

$$A_{\text{GMMAC}} = [24.39 \quad 43.61 \quad 58.08] \tag{6.46}$$

$$A_{\text{GMMAC-filter}} = [22.66 \quad 41.39 \quad 55.90] \tag{6.47}$$

$$A_{\text{GMMAC-controller}} = [21.01 \quad 38.42 \quad 51.83] \tag{6.48}$$

As expected, this design is closest to the MMAC optimized for control and not either of the previous designs for estimation. However, it is interesting that the high end filter extends to a greater frequency than in the cases of the parameter estimation designs.

The GMMAC optimized for parameter estimation was designed according the state approach and yielded the following parameter locations:

$$A_{\text{GMMAC-Param}} = [17.89 \quad 35.71 \quad 52.14] \tag{6.49}$$

$$A_{\text{GMMAC-Filter}} = [16.50 \quad 32.91 \quad 55.06] \tag{6.50}$$

$$A_{\text{GMMAC-Control}} = [15.24 \quad 30.55 \quad 51.06] \tag{6.51}$$

Clearly, these filter locations are significantly different from those in Equations (6.46) through (6.48), which were optimally placed for best control performance. Note that the lowest value for each of Equations (6.49) to (6.51) is substantially lower than the corresponding lowest values in Equations (6.46) to (6.48), as discussed earlier. However, the PFB filters in the design are very close to those for the MMAC and the MMAC optimized for parameter estimation. The MLQG controller in the feedback loop did not have a significant effect on the filter locations.

The results of the performance analysis demonstrate the tradeoff between the optimization for control and the optimization for parameter estimation. Figure 6.23 shows the control performance for the MMAC optimized for control and optimized for parameter estimation. Since the filter parameter locations between the two different MMAC's are different, there will be regions in parameter space in which one controller will outperform the other and vice versa. Though the plots show that the performances appear close, as expected, Table 6.9 does verify that the MMAC optimized for control has the best control performance, according to the performance evaluation tool (to be corroborated through Monte Carlo simulation). However, the MMAC optimized for parameter estimation performs the best for parameter estimation, but it was not anticipated that it would outperform the MMAE. The improvement in parameter estimation performance is only marginal.

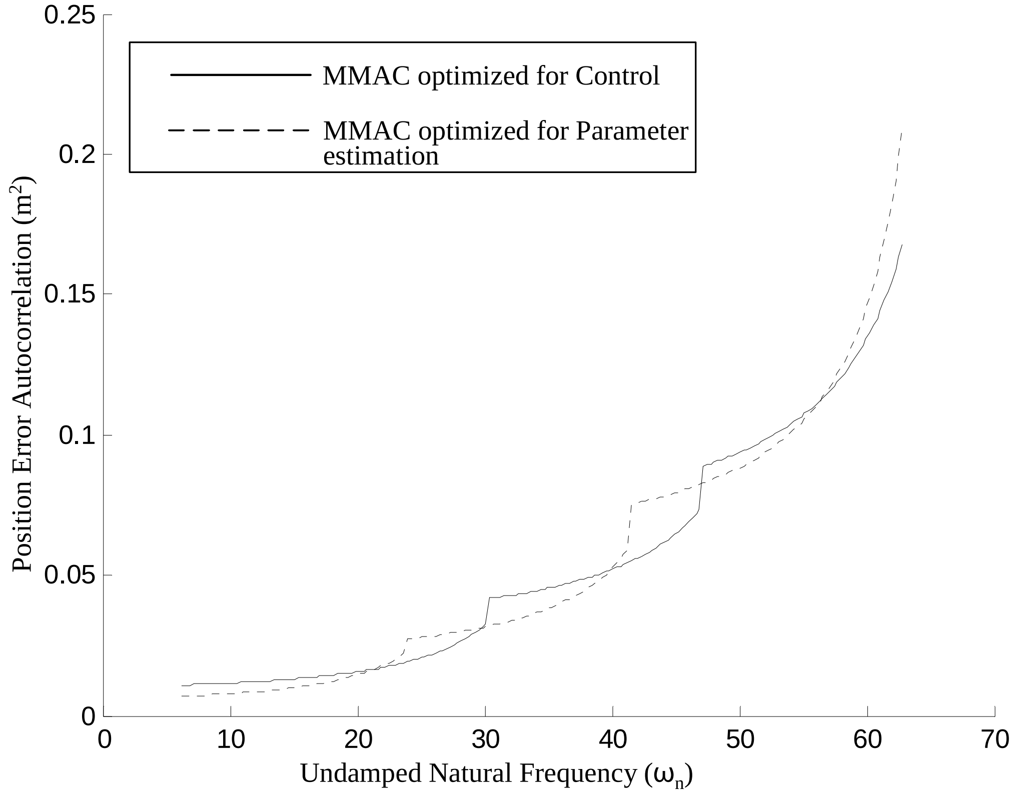

Figure 6.23 Predictive regulation error performance of the MMAC optimized for control and the MMAC optimized for parameter estimation

Controller	Parameter Estimation Predictive Measure Average Mean Squared Frequency (Rad^2/Sec^2)	Control Predictive Measure Average Mean Squared Position ($Meters^2$)
Typical MMAE	36.65	N/A
MMAC-Control	50.13	0.0533
MMAC-Param. Est.	34.30	0.0551
GMMAC - Control	54.87	0.0472
GMMAC – Param. Est.	34.95	0.0483

Table 6.9 Predictive analysis for the parameter estimation and controller regulation performances

Figure 6.24 shows the performance of the MMAC in comparison with the two implementations of the GMMAC. First, since the filter locations for the MMAC and the

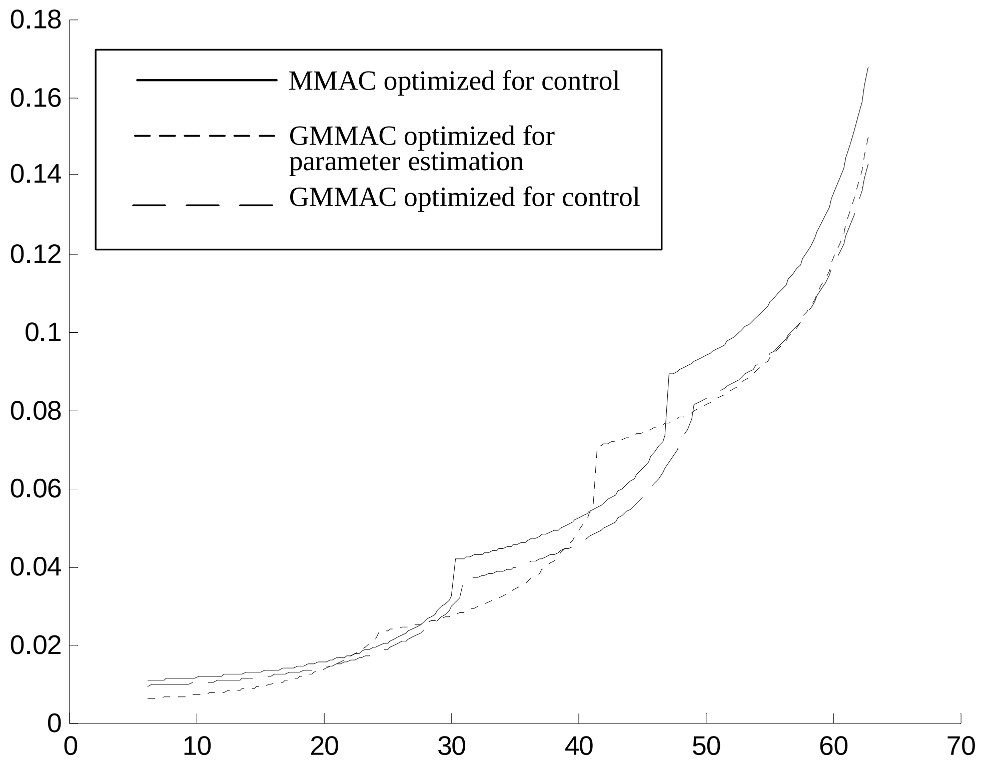

Figure 6.24 Predictive regulation error performance for the MMAC and two GMMAC implementations

GMMAC optimized for control are close, the transition points between filters occur at similar points in parameter space. However, the GMMAC clearly outperforms the MMAC for position regulation. Also, the GMMAC optimized for parameter estimation outperforms the MMAC. However, it has filters at significantly different locations, and thus it performs slightly worse at the filter transition points. The control performance of the two GMMACs is very close, but Table 6.9 indicates that the GMMAC optimized for control will outperform the GMMAC optimized for parameter estimation as expected.

The main goal of this section was to demonstrate the parameter estimation capabilities of the GMMAC. Table 6.9 shows that the predictive parameter estimation performance of the GMMAC optimized for parameter estimation outperforms the MMAE and performs comparably to the MMAC optimized for parameter estimation. For the predicted control performance, the GMMAC discretized for optimal parameter estimation performs superior to the typical MMAC. Overall, the predictive analysis for the GMMAC discretized for parameter estimation indicates that it performs well for *both* parameter estimation and control.

Table 6.10 verifies that the predictive analysis for control is a good indicator of actual performance (as assessed via Monte Carlo simulations), but the parameter estimation actually performs significantly better than predicted. This is true for all the controllers. Figure 6.25 gives a good understanding of the predictive and Monte Carlo analysis results. At the transition points (the peaks of the predicted performance in Figure 6.25) where the predictive analysis gives precise filter transitions, the Monte Carlo analysis tends not to select one filter consistently across the numerous runs. Also, the transition point between filters is not necessarily the average between two adjacent parameter locations (valleys of the predicted performance in Figure 6.25). Thus, selecting the filter not closest in probability, but closet in numerical difference, contributes to reducing the average squared error as shown in Figure 6.25.

Controller	Parameter Estimation Performance from Monte Carlo Simulation Average Mean Squared Frequency (Rad^2/Sec^2)	Control Performance from Monte Carlo Simulation Average Mean Squared Position ($Meters^2$)
Typical MMAE	34.56	N/A
MMAC-Control	43.70	0.0538
MMAC-Param. Est.	26.07	0.0551
GMMAC - Control	46.19	0.0477
GMMAC – Param. Est.	27.20	0.0486

Table 6.10 Monte Carlo analysis for the parameter estimation and controller regulation performances

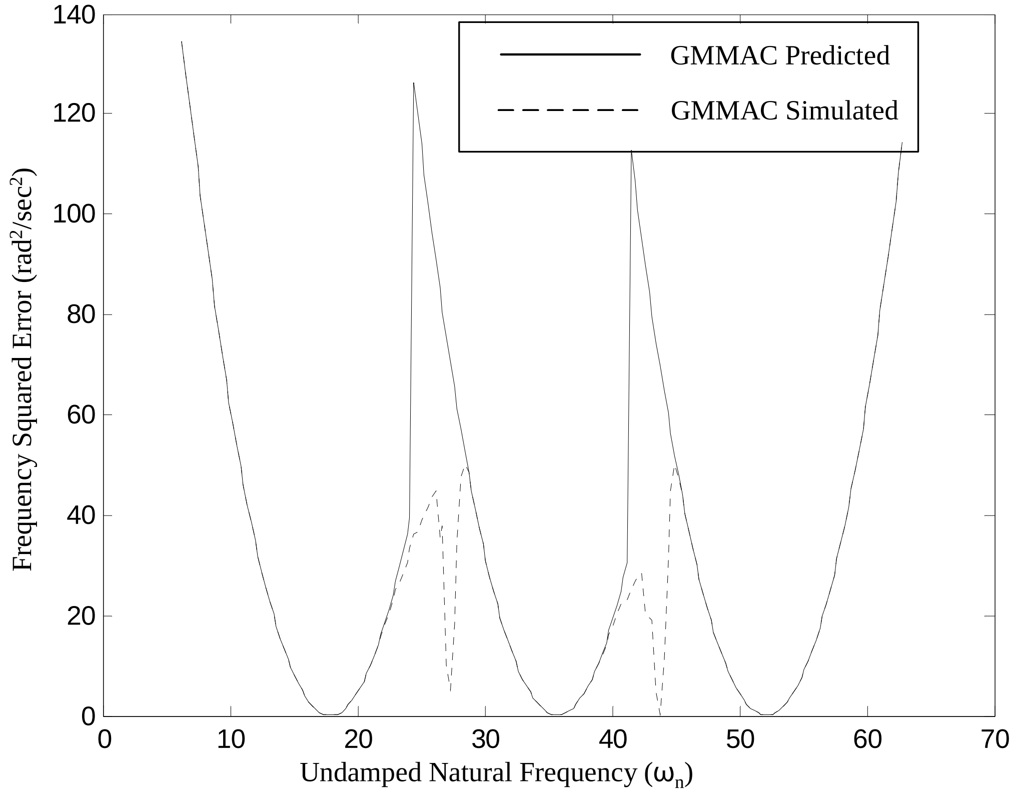

Figure 6.25 Parameter estimation performance of the GMMAC optimized for parameter estimation

This implementation of the GMMAC designed for optimized parameter estimation for the example problem demonstrates that there is a small sacrifice in control performance in comparison to the GMMAC optimized for control. However, the GMMAC can be used to satisfy the requirements for parameter estimation and control simultaneously. To satisfy these same requirements otherwise would require implementation of both an MMAE and an M^{3}AC. However, it must be considered that the resources to implement the GMMAC are less than those required for the MMAE and the M^{3}AC combined. The engineer has the tools to decide if the performance of the GMMAC is sufficient in comparison to the resource savings.

6.5 MMAE-Based Control

This section investigates the application of the MMAE-based control techniques developed in Chapter 5 to the ideal mechanical-translational system described in Section 6.1. It was established in Chapter 5 that the MMAE-based control and the MMAC have equivalent structures at steady state (assuming no lower bounding on the elemental controller, though not true in practical implementations). Thus, the design procedures for the proposed architectures are very similar to the MMAC techniques. Results for the predictive analysis and Monte Carlo simulations will be compared with the previously assessed MMAC and LQG approaches where appropriate. Whereas the MMAC results concentrated on the steady state analysis, MMAE-based control will also include the transient response. There lies any potential improvement of the MMAE-based control over the MMAC.

The different MMAE-based control architectures discussed in Chapter 5 will be evaluated. Each offers a slightly different method of obtaining the desired performance and degree of complexity. For example, the simplest of the architectures is the MMAE-based control with a nominal gain. Although most likely not the design of choice, it is worth investigating to determine a baseline for the more complex MMAE-based designs. At the other end of the spectrum, the most complex architecture entails the LQG controller replacing the controller element.

6.5.1 MMAE-Based Control with a Nominal Control Element

The basic form of the MMAE-based control involves the state estimate feeding a nominal controller. This would be equivalent to an MMAC architecture in which the gains in the elemental LQG controller gains are exactly the same. Since there is only one controller gain, the parameter estimate typically used to select the controller is unnecessary. This section evaluates the implementation of this approach and compares it against the typical MMAC design.

6.5.1.1 MMAE-Based Control with a Nominal Control Element Design

As in the previous examples, there are three elemental filters that have to be placed in parameter space, but there is only one controller gain to be determined. Thus, there are four variables that specify the design, which can be determined easily using an optimization routine. A basic MATLAB optimization routine *fminsearch* [31], was used to determine the optimal filter placement of the filters and the single controller gain that minimizes the average position correlation across the parameter space. The resultant filter and controller parameters are given by:

$$A_{\text{nom-filter}} = [21.01 \quad 43.25 \quad 51.64] \tag{6.52}$$

$$a_{\text{nom-controller}} = [47.91] \tag{6.53}$$

The average position correlation of this filters/controller combination is 0.0669 m^2. A comparison with the performance results in Table 6.8 from the previous section, clearly demonstrates that the single controller element approach does not perform as well the MMAC approaches.

6.5.1.2 MMAE-Based Control with a Nominal Control Element Results

The results for the MMAE-based control with a nominal control element demonstrate the effects of the filter placement in the parameter space and the tradeoffs in performance. Also, it is clear that the MMAE-based controller with a single nominal controller gain does not offer performance as good as the typical MMAC.

To illustrate the effects of the filter placement, Figure 6.26 presents an overlay of the predicted performance for two different placements of the filters in parameter space with the corresponding controller gains. The first MMAE-based controller is specified in the previous section and the second is given by:

$$A_{\text{nom-filter}} = [20.85 \quad 29.27 \quad 51.53] \tag{6.54}$$

$$a_{\text{nom-controller}} = [47.82] \tag{6.55}$$

This second design was a local minimum during the minimization of the position correlation. The average position correlation with this second filters/controller

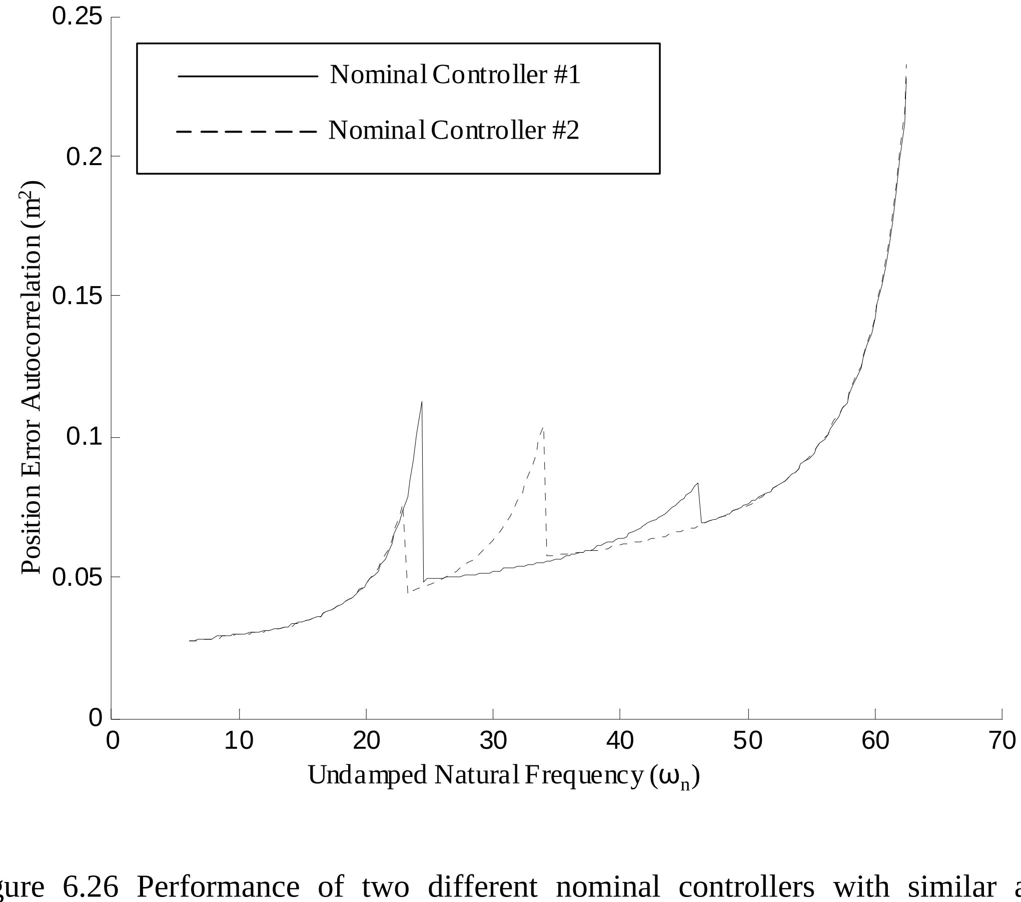

Figure 6.26 Performance of two different nominal controllers with similar average position correlation over the parameter space

combination is 0.0673 m^2, which is clearly not significantly different from the previous design. However, the plots of the resultant performance evaluation are very different.

For both nominal controllers, the plots of the predicted performance across parameter space indicate where the MMAC switches from one filter to the next. Not only are the plots discontinuous, but the plot of the cost function indicates a substantial step reduction in position error at the switch points. Clearly, the single controller limits a smooth transition from one filter to the next. As the performances of the two controllers demonstrate, the amount of the jump is affected by the relative positions of the filters in parameter space. There is a tradeoff between a larger relative jump at one transition point versus a smaller jump at a different transition point. It is possible that two rather small jumps at the transitions can yield the same overall average performance over the parameter space as the case of a larger and a smaller jump.

Both nominal MMAE-based controller implementations also demonstrate that the overall cost curve is greater than that for the typical MMAC from the previous section. Clearly, this is a result of the limitation of a single controller gain. The model on which the controller gain is based is closer to the filter model with the larger ω_n than the other two models. The larger gains associated with the models based on the smaller values of ω_n would drive the response unstable for larger system values of ω_n. Hence, the controller model parameter value is determined by the requirements of the high end of the parameter space.

The Monte Carlo simulation of the first controller implementation shown in Figure 6.27 indicates that there is a problem with the prediction at the filter crossover

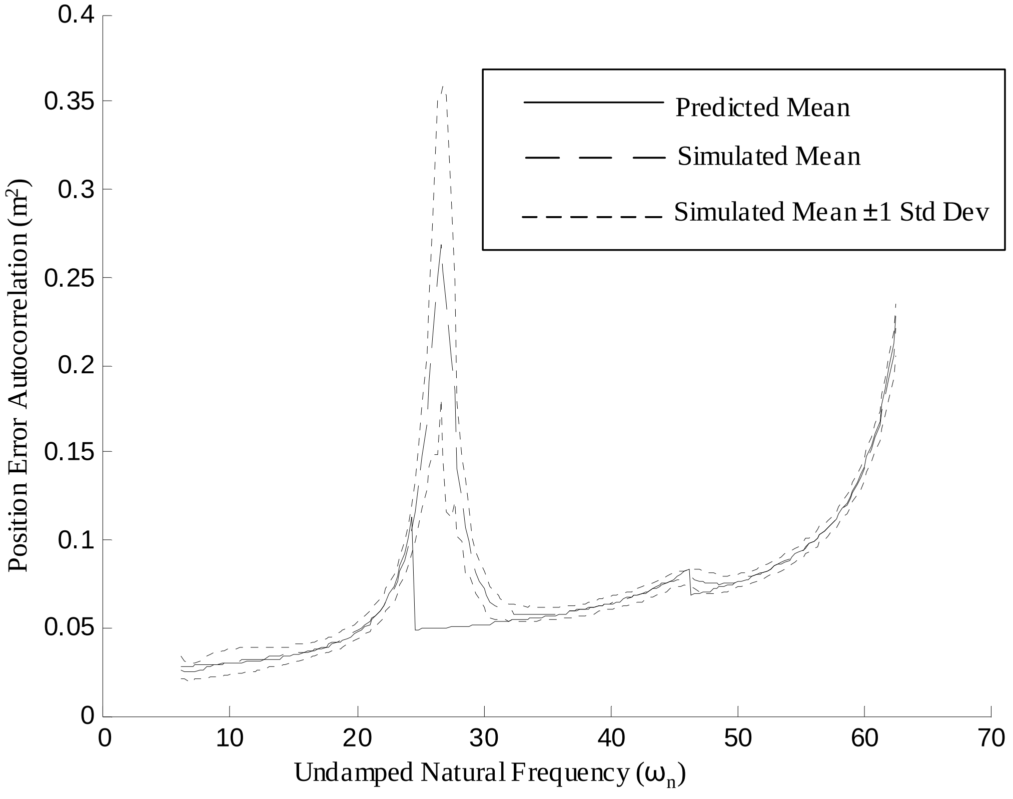

Figure 6.27 Monte Carlo simulation overlaid predicted performance for the first nominal controller

point. Clearly, the correct filter is not selected consistently until the value of ω_n for the system parameter is considerably greater than the predicted transition point.

Ironically, the Monte Carlo simulation of the second controller, shown in Figure 6.28, was significantly closer to the predicted performance than it was for the first controller. As indicated in Table 6.11, the second implementation which had the slightly higher predicted cost, actually has better performance as indicated by the Monte Carlo analysis. However, both implementations perform more poorly than the baseline MMAC. For the second implementation, the predicted performance at the transition

Controller	Predictive Measure Average Mean Squared Position (Meters^2)	Monte Carlo Simulation Average Mean Squared Position (Meters^2)
Baseline MMAC	0.0533	0.0538
Nominal Controller 1	0.0669	0.0785
Nominal Controller 2	0.0673	0.0731

Table 6.11 Results for the Nominal MMAE-based Control

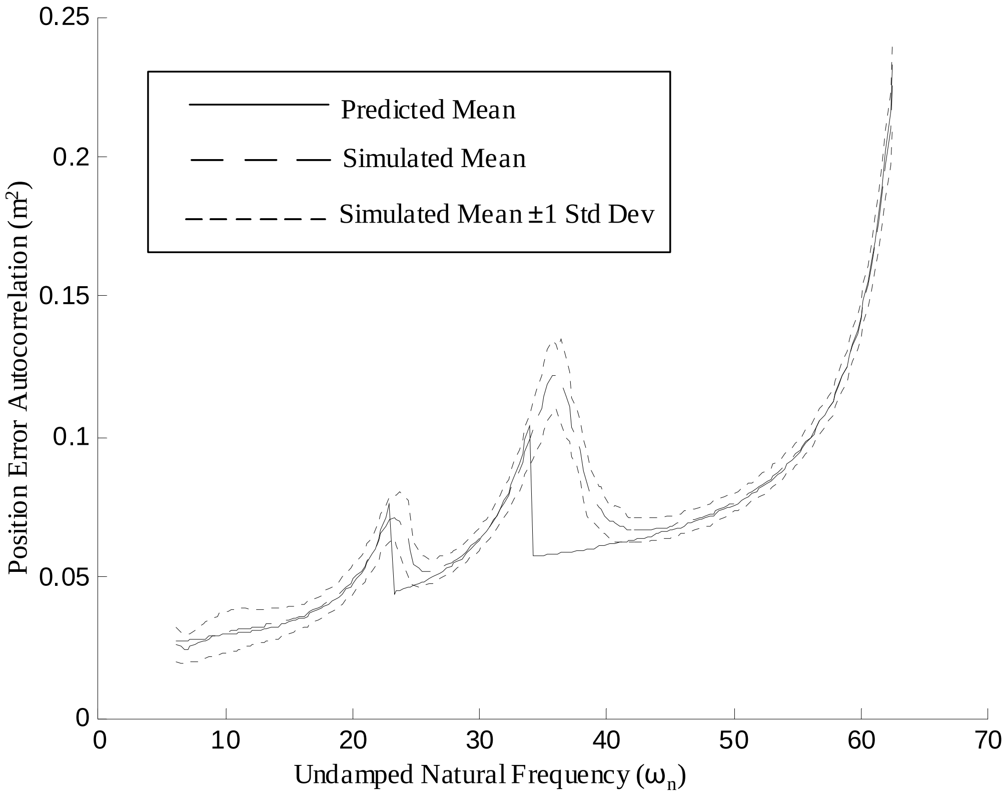

Figure 6.28 Monte Carlo simulation overlaid predicted performance for the second nominal controller

points is still not very accurate. The main difference for the second implementation is that the poor selection of the filter does not have as significant an increase in the position error at the second transition point as the first implementation had at the first transition point. For the first design, there is a large region around the first transition point where the incorrect filter selection causes poor performance.

The effects of the transitions can be demonstrated by the covariance proximity measure used in the filter selection developed in Chapter 4. The proximity measure and filter standard deviations for the three different filters for the first design are plotted in Figure 6.29. As discussed in Chapter 4, the minimum of the three proximity measures will be the one associated with the filter that is selected. The one standard deviation envelopes indicate that that minimum of the proximity measure may not be an absolute transition when the variance of the measure is considered. The upper bound on each mean plus one standard deviation plot goes to infinity while the lower bound on the mean

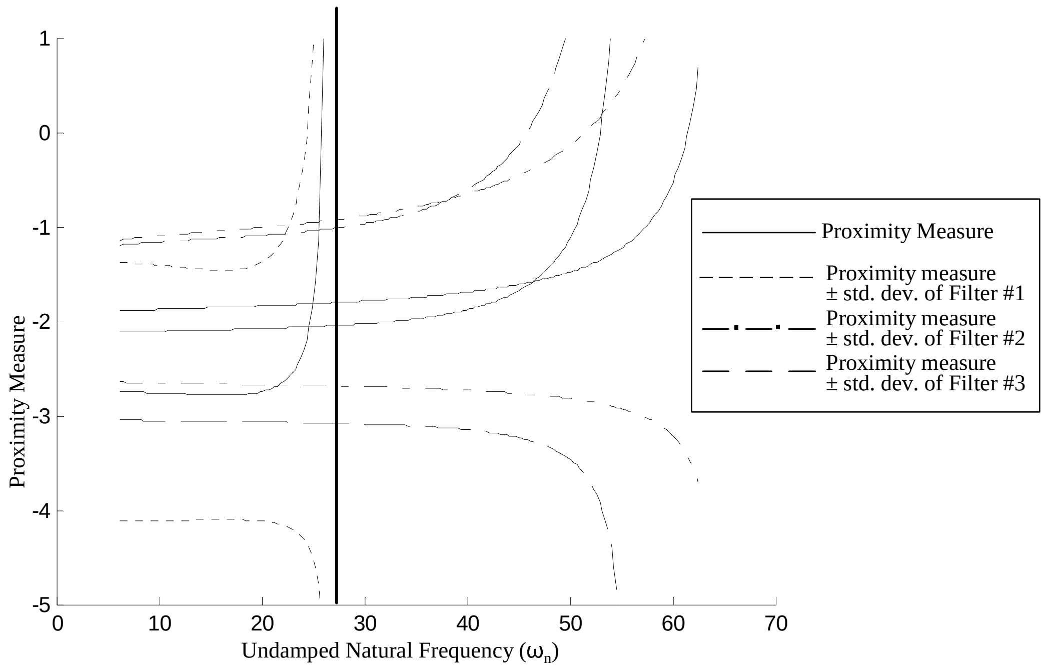

Figure 6.29 Predicted proximity measures and their standard deviations of the individual filters for the first nominal controller design

minus one standard deviation goes to minus infinity. The bold vertical line in Figure 6.29 indicates the asymptotic bounds of the first filter. Observe that this bold line is located past the crossover point of the proximity measures for the first and second filter. Thus, as demonstrated with the Monte Carlo simulation, the transition between the two filters does not always occur at precisely the same ω_n values. The variance in the transition point further demonstrates that the incorrect filter may be selected, and thus the degradation in performance.

6.5.2 MMAE-Based Control with a Blended Controller Element

This MMAE-based control approach blends controller gains according to probability weighting on the filters in the filter bank into a single gain that is then multiplied by the state estimate from the MMAE portion to produce the final control. As stated in Chapter 5, this is a refinement over the MMAE-based control with a nominal controller, but it does not offer any significant improvement over the typical MMAC. This implementation is simply a probability weighting of the possible gains with a post-multiplication of the state estimate, whereas the MMAC involves a multiplication of the state estimate by the controller gain with a probability weighting applied to each individual control.

6.5.2.1 MMAE-Based Control with a Blended Controller Element Design

The design steps for this architecture are exactly the same as that for the M^{3}AC. The number of filters determines the number of controllers that will be blended together to produce the final controller gain. The steady-state response will determine the filter locations. As for the design of elemental controllers, the M^{3}AC approach which uses the

MLQG controllers yields the best architecture in terms of performance over the typical MMAC.

Since the design steps are the same as the M^{3}AC, the resultant design from that implementation is repeated here. The parameters for the design of the filters are given by:

$$A_{\text{blend-filter}} = [24.47 \quad 43.46 \quad 56.28] \tag{6.56}$$

The corresponding controllers are designed for the models based on parameter ω_n given by:

$$A_{\text{blend-controller}} = [22.70 \quad 40.37 \quad 52.18] \tag{6.57}$$

Again, the design approach assumes that one filter and controller will be selected at steady state. It does not take into account any differences between the probability weighting of the possible gains with a post-multiplication of the state estimate versus the multiplication of the individual filters' state estimate by the controller gain with a probability weighting applied to each individual control. The impact of those effects will be exposed in the Monte Carlo simulations of the next section.

6.5.2.2 MMAE-Based Control with a Blended Controller Element Results

The Monte Carlo simulation plotted in Figure 6.30 shows empirically that the MMAE-based control with a blended controller element is not equivalent to the M^{3}AC. However, the differences in performance are only at the transition points. Not only does the blended control have greater position error compared to the results for the M^{3}AC shown in Figure 6.20, the standard deviation bound is significantly greater as well. The results for the MMAC and MMAE-based controller with blended control shown in Table 6.12 indeed indicate that the predictive measures are the same, but the Monte Carlo simulations differ by about one percent. Again, this insignificant difference comes from the performance at the transition points between the filters.

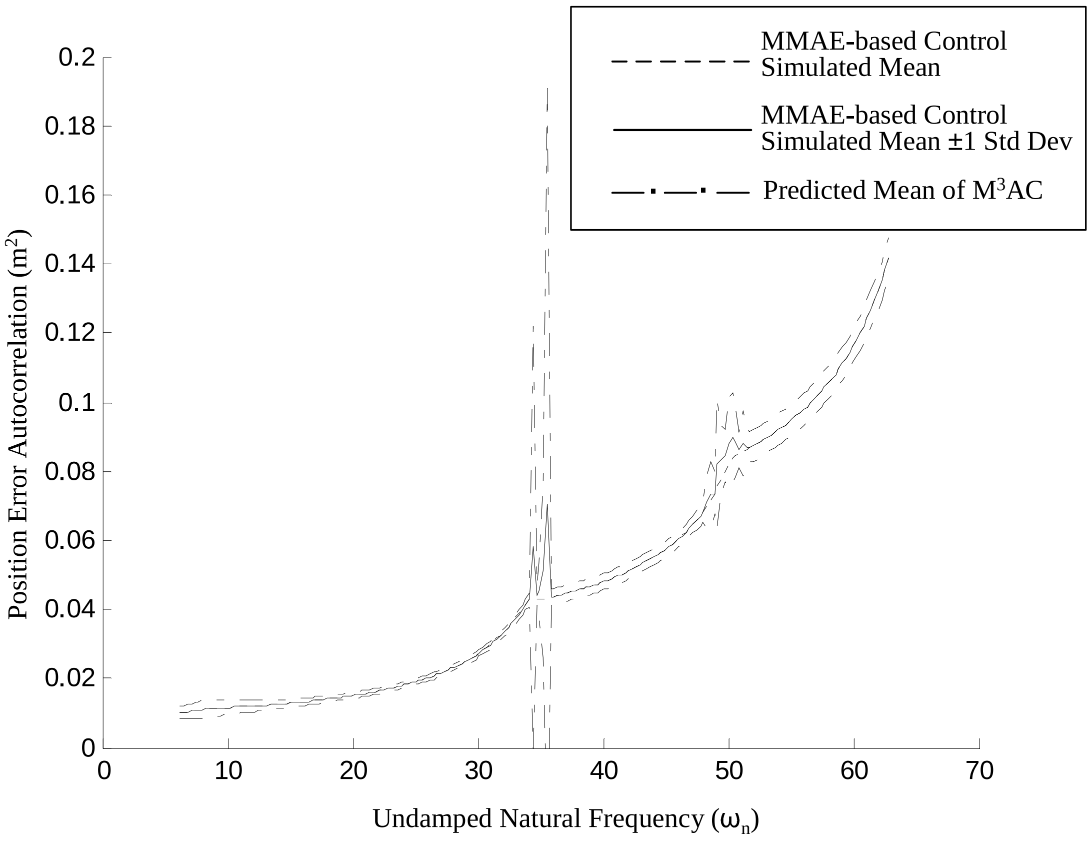

Figure 6.30 Monte Carlo Simulation for the MMAE-based control with blended controller gain and the M^3AC

Controller	Predictive Measure Average Mean Squared Position (Meters^2)	Monte Carlo Simulation Average Mean Squared Position (Meters^2)
M^3AC	0.0472	0.0476
MMAE with Blended Control	0.0472	0.0482

Table 6.12 Results for the MMAE-based controller with blended control compared with M^3AC results

6.5.3 MMAE-Based Control with a Selected Controller Element

This section evaluates the MMAE-based control architecture that uses the parameter estimate to select the controller gain that will be used in the control computation. The MMAE substructure provides not only the state estimate, but also the parameter estimate. The parameter estimate is then used as index for a look-up table of controller gain values. Recall that, in the review of MMAE-based control in Chapter 2, it is established that the MMAE can not be designed simultaneously for best parameter estimate as well as best state estimate. Of course, the best state estimate translates into the best control. Thus, locations in parameter space for the filter design are set for best control and can be taken directly from the M^{3}AC. Now in order to compensate for the fact that the parameter estimate may not be the *best*, the controller gains are designed based on the individual probabilities that make up the parameter estimate as discussed in Section 5.2. As is the case of the previous multiple model analysis designs, the controller's capability will be limited by the performance of the MLQG controllers evaluated at points across the parameter space as given in the results of Section 6.3.

6.5.3.1 MMAE-Based Control with a Selected Controller Element Design

As discussed in Chapter 5, the design of the MMAE-based control with a selected controller element is essentially a two-step process. First, the elemental filter placement for the MMAE portion of the architecture is accomplished by discretizing the parameter space for best control. Clearly, this will yield the same discretization as the M^{3}AC design. The results of the M^{3}AC discretization from Section 6.4 yield:

$$A_{M^3AC\text{-filter}} = [24.47 \quad 43.46 \quad 56.28] \tag{6.58}$$

The second step in the design process is to determine the controller gains for the look-up table with which the parameter estimate is used as the index into the possible controller gain values. Of course, for this example that has three filters in the MMAE portion of the architecture, the parameter estimate is given by:

$$\hat{a} = p_1 a_1 + p_2 a_2 + p_3 a_3 \tag{6.59}$$

where p_1,p_2, and p_3 are the probability weights for each filter and a_1, a_2, and a_3 are the parameter values for the filter designs. For the implementation, the discrete values of probabilities were evaluated in the possible range of zero to one with an interval of .01. Of the possible 1,000,000 probability vectors, only 3,468 were allowable because they also had to meet the requirement that

$$p_1 + p_2 + p_3 = 1 \tag{6.60}$$

as well as the logical condition

$$\text{NOT (p1>p2 and p3>p2)} \tag{6.61}$$

Equation (6.61) simply addresses the physical condition that the parameter estimate must exist between the parameters that specify the adjacent filters.

Of those combinations that were viable, the probabilities were mapped back to the possible discrete values of $\hat{a}$ that make up the look-up table. Clearly at the extreme values of the parameter space, only one probability vector makes up each parameter estimate, e.g. [1 0 0] and [0 0 1]. However, the interior parameter estimate is not only generated by the parameter estimate [0 1 0], but also other combinations satisfying Equation (6.60) and Equation (6.61).

Now the possible probability vector combinations are used to determine the controller gain values for each $\hat{a}$. For a specified controller gain and each possible probability vector, the performance was computed with the probability lower bound using equations from Section 5.3.3.2 and the system model parameters assumed to be equivalent to $\hat{a}$. The next step is to compute the RMS error of the resultant performances. The MATLAB optimization routine *fminsearch* [31] was used to determine the controller gain that would minimize the computed RMS error value.

The resultant controllers are shown in Figure 6.31, overlaid with MLQG controllers that are designed for each point in parameter space. Note that the controller gains are constant before the first and after the last filter parameter locations, respectively. Those controller values are set to the controllers for the M^3AC for the cases in which $\hat{a}$ is less than the first filter location and greater than the last filter location, respectively. However, a value for $\hat{a}$ that is less than the first and greater than last filter locations is not a physically possible condition in a multiple model structure (since $\hat{a} = \sum_{k=1}^{K} a_k \cdot p_k$), but the controllers are included for completeness. Also note that the controllers around the center of the parameter space are based on ω_n greater than the frequencies used for the corresponding MLQG designs. This indicates that the controllers for the MMAE-based control design are much more conservative at those middle points of the parameter space. Rather than assuming a perfect estimate for $\hat{a}$, the design accounts for possible probability weights that could form the $\hat{a}$. This will include the effects of all the filters outputs rather than just an ideal filter at the value of $\hat{a}$ as is used for the MLQG controllers in Figure 6.31.

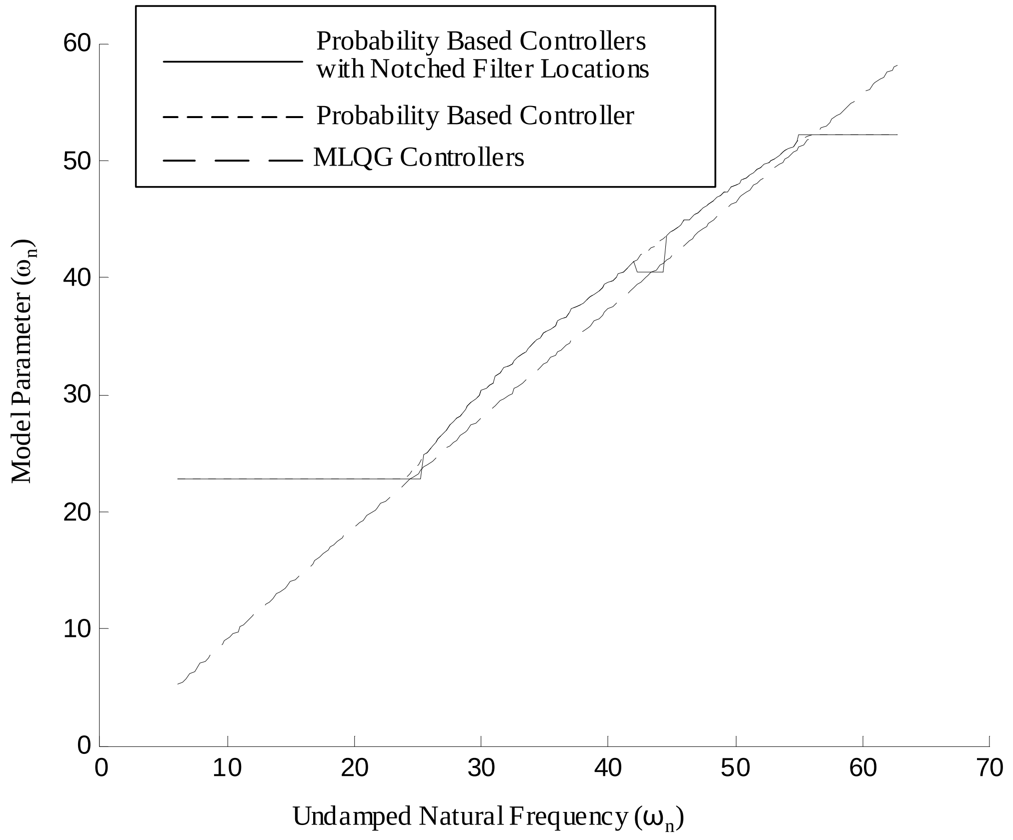

Figure 6.31 Controller gains for the three different look-up controller gain tables

The second design simply takes the results from the previous step and, for a small region (or notch) around the filter locations, sets the controller to be equivalent to the M^{3}AC filter value. The resultant controllers are plotted in Figure 6.31 along with the previous design. The controller value across the entire notch is equivalent to the value at the center of the notch which is equivalent to the M^{3}AC filter location. The controllers for the M^{3}AC are equivalent to the MLQ controller at the specified filter location. As previously discussed in Chapter 5, the MMAE-based controller will settle to a single filter at steady state. Since the each elemental controller designed for M^{3}AC has been optimized for the best performance, it is assumed that the MMAE-based controller will duplicate this performance at steady state. The transient response will be affected by the controllers that are designed for the parameter estimates that are not close to the filter locations. These parameter estimates are outside the notch shown in Figure 6.31. It will be the controllers corresponding to these parameter estimates that will drive the system to steady state.

6.5.3.2 MMAE-Based Control with a Selected Controller Element Result

The designs from the previous section yield three different implementations of the MMAE-based control with a parameter estimate look-up table to be tested and compared. The three implementations use the parameter estimate as the look-up table index to find the controller gain. The three controller gain design approaches were the MLQG-based designs, the probability vector derived controllers, and the *notch* design. The only difference among the designs is the contents of the look-up tables of controller gains. Each table of gain values has the same number of elements and the computation of the parameter estimate that points into the table is also the same for testing all the designs. Each design will be compared to the M^3AC Monte Carlo Analysis.

Figure 6.32 shows the predicted performance for the two different controller look-up tables' designs overlaid with the predicted performance of the MLQG. The predicted performance of the MLQG across the parameter space is equivalent to the predicted results of the third look-up table design. The controller gains are equivalent to those designed for the MLQG controller at each assumed point in parameter space. Interestingly, in a small region above the third filter parameter value, the parameter estimation design approach found a controller gain that gave a slightly better performance. The controller gain was forced to a lower model value corresponding to the

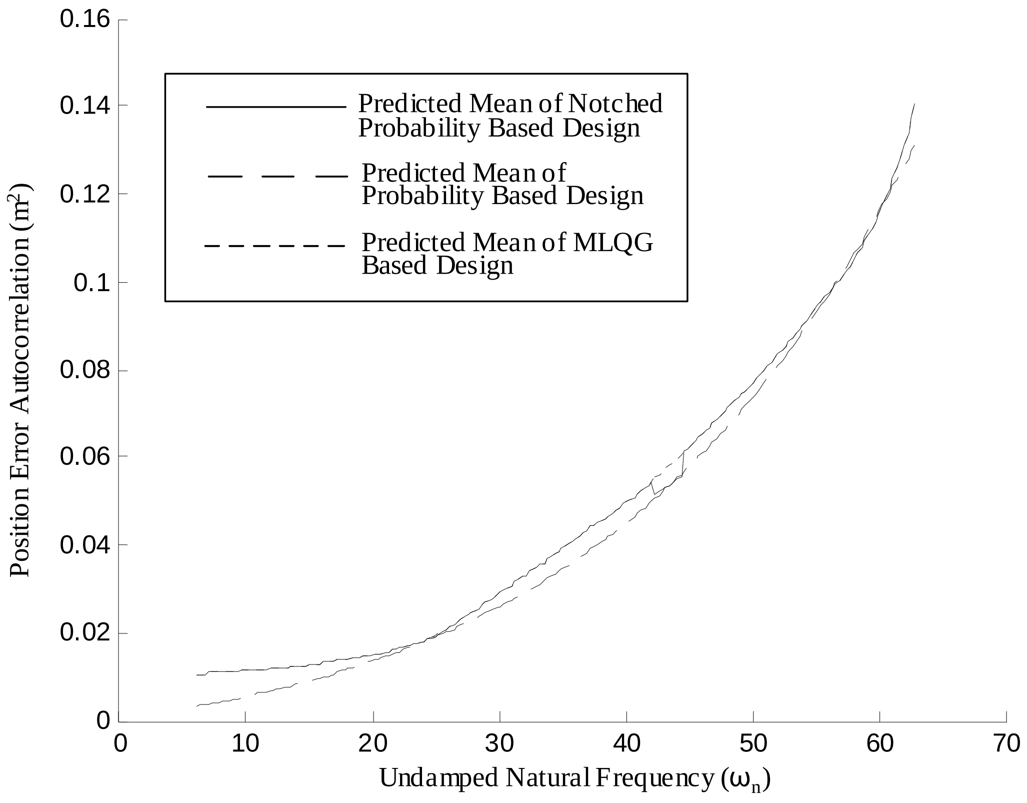

Figure 6.32 Predicted performance of the three controller look-up table designs

last filter model. The small difference in controller gains due to the resolution of the MLQG controller optimization for the few subsequent assumed parameter estimates is the reason for the slight improvement.

The first controller design based on an assumed parameter estimate, also takes into account the possible probability vectors that form the parameter estimate. Compensating for all possible probability vectors where more than one parameter vector forms the assumed value of the system, yields a more conservative design. As shown in Figure 6.32 for the parameter placement in Equation (6.58), the predicted performance in the region around the middle filter does not have the same performance as if the controller were designed only for the perfect parameter estimate. However, at either of the two end filter locations, only one probability vector forms the parameter estimate, and thus the predicted performance is the same as that for MLQG controller.

The resultant performance of the second design, referred to as the notched design, is a mix of the first design and the MLQG controllers. As shown in Figure 6.32, in the regions around the filters' locations in parameter space where the controller gains were set to the MLQG controller gains, the performance is equivalent to the MLQG controllers. At all other locations, the performance is equivalent to the first design.

The predicted performances shown in Figure 6.32 assume that the MMAE has perfectly predicted the parameter estimate at steady state. This gives only an idea how the filter would perform before it reaches steady state. At steady state, only one filter with the corresponding parameter location will be selected and it will not necessarily be equivalent to the true system. Hence, the Monte Carlo analysis is useful to reveal how each design actually behaves at the filter selection transition points and regions of the parameter space where there is the greatest mismatch between the filters and the assumed true system.

Table 6.13 shows the Monte Carlo analysis of the design based on the MLQG controllers with look-up gain table, overlaid with the simulated mean of the M^{3}AC. Although this MMAE-based implementation matches the M^{3}AC performance very well, it does not offer an improvement. In fact, at the filter transition points, there is a slight degradation in performance. As far as improvement to the transient response, results for the average of the mean position correlation over the initial transient region shown in Table 6.13 indicates that M^{3}AC outperforms the MMAE-based control with MLQG controllers.

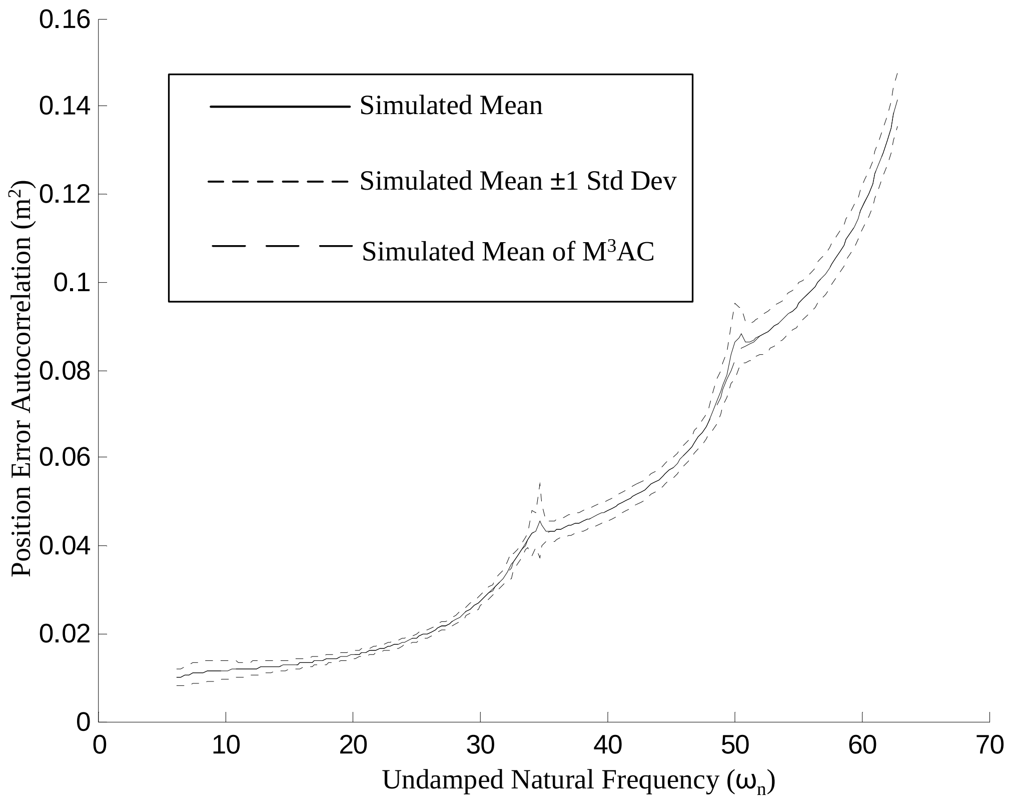

Figure 6.33 Monte Carlo analysis of parameter estimate selected controllers based on MLQG models overlaid with predicted mean of the M^3AC

Controller	Monte Carlo Simulation (Steady State) Average of Mean Squared Position (Meters2)	Monte Carlo Simulation (Transient) Average of Mean Squared Position (Meters2)
M3AC	0.0476	0.0483
MMAE-based, MLQG controllers	0.0479	0.0494
MMAE-based, using probabilities	0.0493	0.0498
MMAE-based, notch	0.0479	0.0489

Table 6.13 Results for the M^3AC with robustness compared to the MMAE-based controllers with controller gains selected via parameter estimation methods

Using the possible probability vectors that contribute to the parameter estimate clearly does not enhance the MMAE-based controller performance, as indicated in Figure 6.34. Over most of the parameter space, the controllers that were optimized for all possible probability combinations for the parameter estimate did not perform as well as the M^3AC or the previous design approach. As the predicted analysis confirmed, design for all the possible probability vectors that form parameter estimate degrades the performance even at the middle filter where the filter model matches the parameter estimate. This indicates that the controller really does not have to be designed to protect

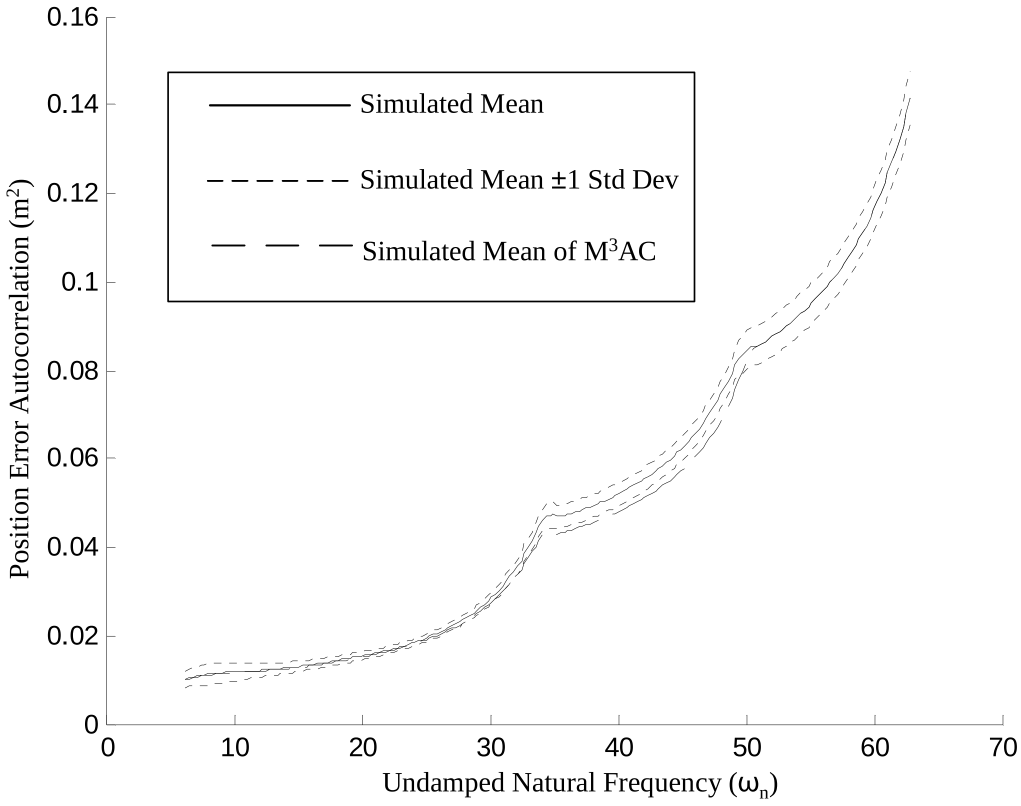

Figure 6.34 Monte Carlo analysis of the MMAE-based controller designed for possible probabilities that form the parameter estimate overlaid with the predictive analysis for the M^3AC

against all possible probability vectors that form the parameter estimate. That is too conservative. Further, Table 6.13 indicates that the controller design did not improve the transient performance either. This result, along with the steady state response results, indicates that for this example problem, the effects of any errant effects of the probability vectors that form the parameter estimates are minimal.

The final design tested is the *notch* design, which is a composite of the previous two tested designs. The Monte Carlo analysis shown in Figure 6.35 indicates the results are very similar to those for the MLQG controllers of the first design tested. This

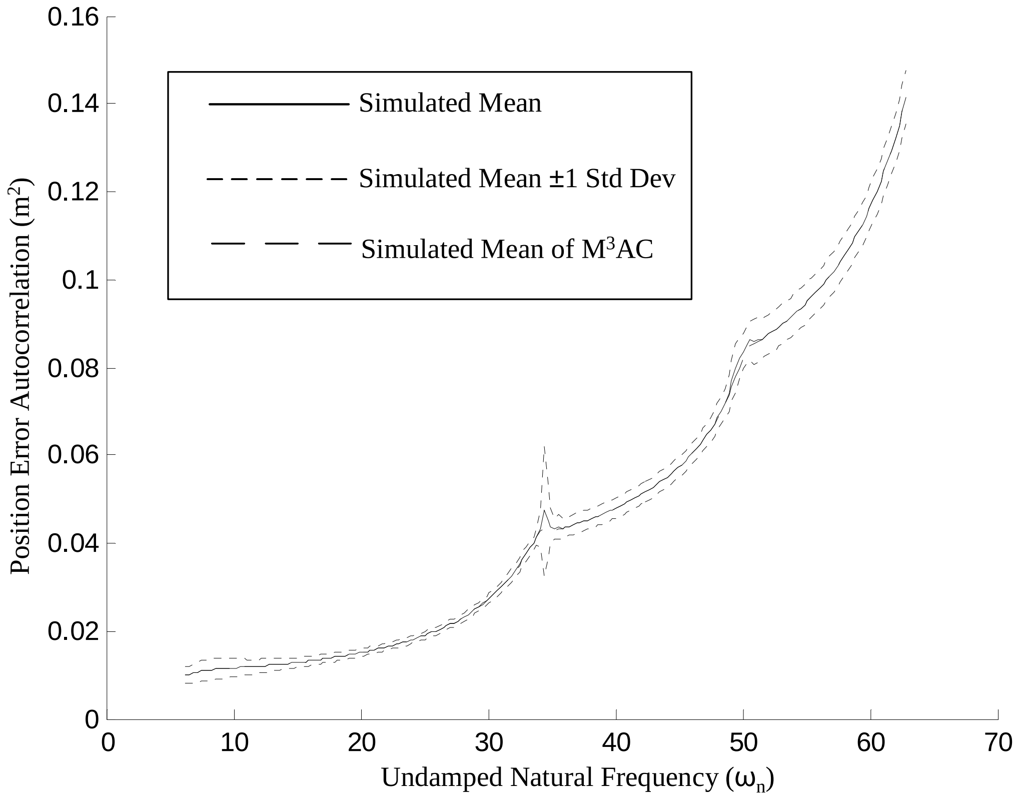

Figure 6.35 Monte Carlo analysis of notched probability look-up controller table overlaid with M^3AC predictive analysis

indicates that the parameter estimate quickly settles to the filter closest in parameter space and that this transient does not have a great effect on the overall performance. The results summarized in Table 6.13 indicate a slight improvement over the previous designs, but not of great significance. This further confirms that the parameter estimate quickly settle to the closest filter according to the Baram proximity measure. Transient effects of the parameter estimate are minor.

Overall, for the three designs, analysis indicates that for this example problem, the effects of probability vector variations that occur before steady state has been reached are minimal. The simple design of selecting the MLQG controllers corresponding to the possible parameter estimates outperformed the more complex probability vector based design. However, not one of the three designs outperformed the M^{3}AC design. The blending of the control performed in the M^{3}AC better reflects the control required at any given operating point of the system.

6.5.4 MMAE-Based Control with Probability-Based Table Look-up

The MMAE-based control with probability-based table look-up is the general case of the previous controller in which the controller gain is selected based on the computed parameter estimate. Since more than one parameter vector may represent the parameter estimate, information provided by the parameter vector is lost in forming the parameter estimate. However, where the amount of information may be an advantage for the probability vector look-up table approach, the size of the table is clearly a disadvantage. In the previous example, 3,468 valid probability vectors were reduced to 200 possible parameter estimates. For each valid probability vector, a controller gain matrix must be designed.

6.5.4.1 MMAE-Based Control with Probability-Based Table Look-up Design

The MMAE-based control with a probability-based table look-up is a general case of the parameter estimate table look-up approach and the design process is simple. The same set of admissible probability vectors from the previous case is used to build the look-up table. For each probability vector, the associated parameter estimate as computed in Equation (6.59) is considered the parameter of the true system. Now, for the probability vector and the assumed true system, MATLAB's *fminsearch* [31] optimization routine is used to find the controller that minimizes the position correlation that is computed using the probability lower bound equations from Section 5.3.3.2. This process is repeated for every admissible probability vector.

As was the case for the previous designs, the probability vectors [1 0 0], [0 1 0], and [0 0 1] will have controllers that correspond to the M^3AC designs. This of course assumes that a lower bound is not placed on the filters in the MMAE portion of the architecture, though in practice a small lower bound is used to prevent filter lockout. Hence, the same parameter locations as the M^3AC are used as the filter locations in the MMAE portion of the architecture and are given as:

$$A_{\text{MMAE-filter}} = [24.47 \quad 43.46 \quad 56.28] \tag{6.62}$$

Since there is not a one-to-one correspondence between the assumed true system models and the controller models, it is not easy to compare this with previous approaches. Assuming that each probability vector can form a parameter estimate of the assumed true system, then the controller models for the assumed true system can be plotted as shown in Figure 6.36. The solid region delineates the 3,486 controller models that correspond to the design for the possible parameter vectors. The probability vector is used to compute

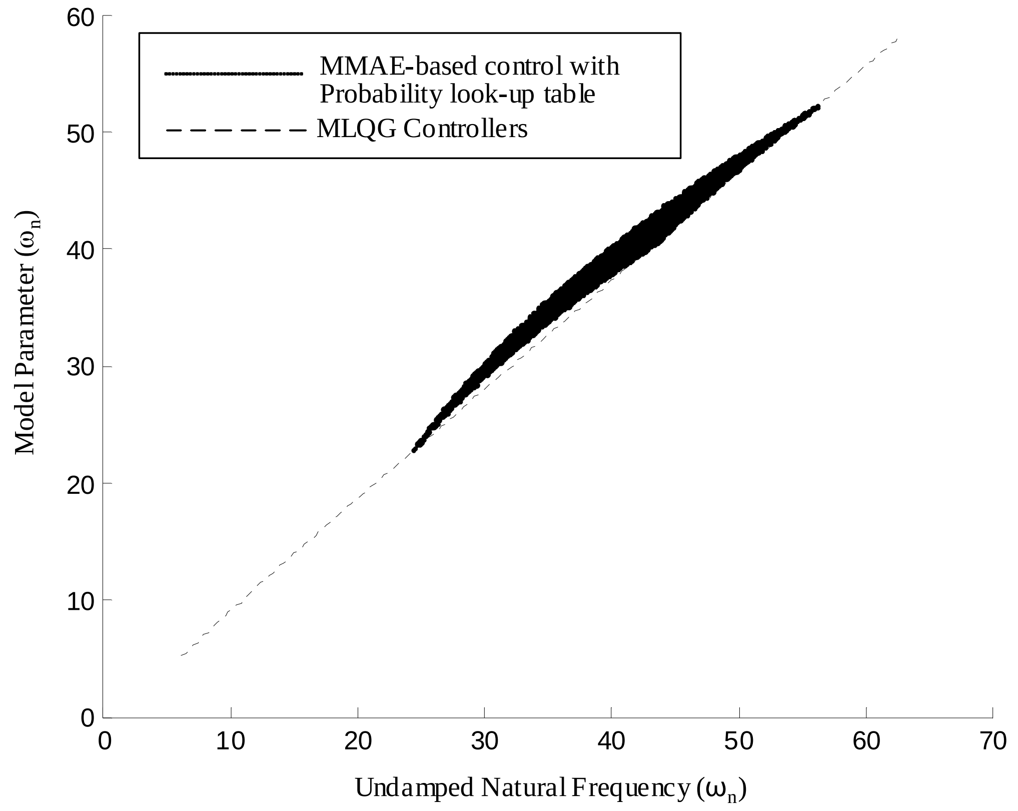

Figure 6.36 MLQG controller and table look-up controllers for assumed values of $\hat{a}$

the parameter estimate that corresponds to the controller. As expected, the region of probability based controllers touches the plot of the MLQG controllers at the points in parameter space that correspond to the filter locations. Those points correspond to the probability vectors [1 0 0], [0 1 0] and [0 0 1]. This is an indication that the performance should at least duplicate that of the MMAE-based control using MQLG controllers as analyzed in the last section.

6.5.4.2 MMAE-Based Control with Probability-Based Table Look-up Results

The predictive analysis and Monte Carlo simulation results are presented in similar manner as for the controllers in the previous section. For the predictive analysis, the performance of the controller is computed for each probability vector only at the point given by the associated parameter estimate. It is assumed that the associated parameter estimate found by the MMAE accurately matches the true system parameter. The resultant performance of the 3,468 controllers is presented in Figure 6.37 and appears as an envelope of performance as discussed in Section 5.3.2.4. This envelope covers all possible conditions during the controller operation and thus describes the predicted performance.

Similar to the controller plots of the last section, the MLQG controllers evaluated across the parameter space touch the region of predicted performance at the three filter

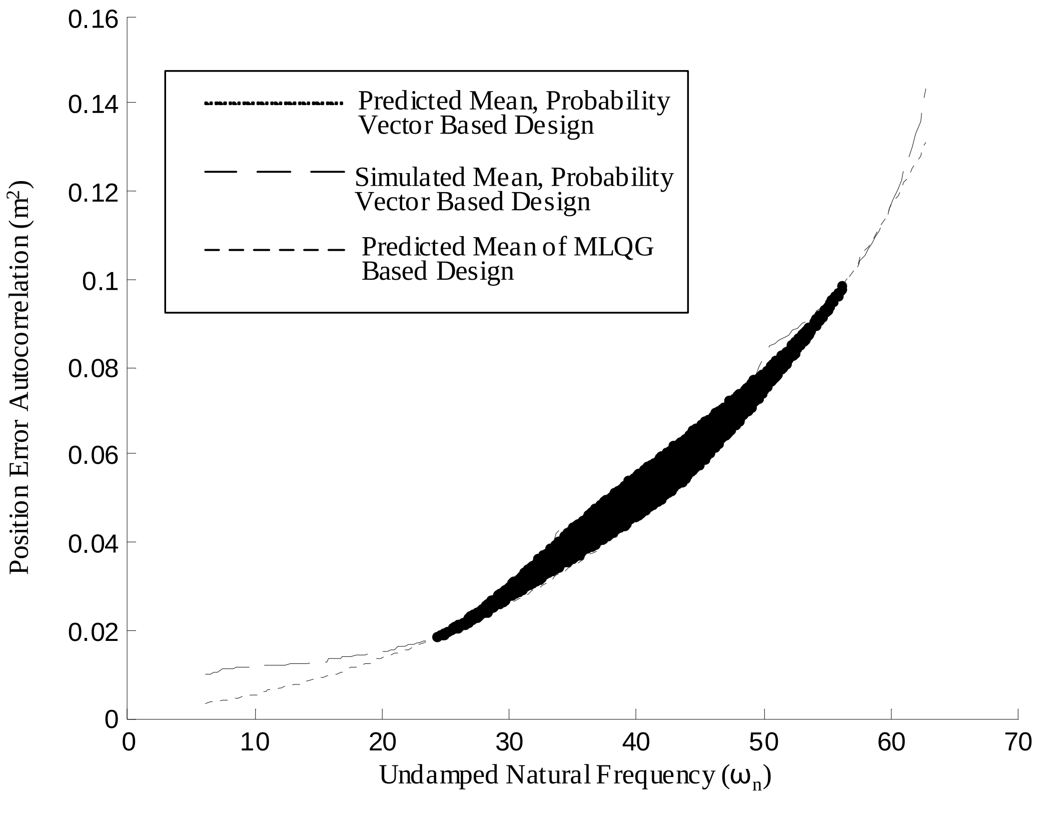

Figure 6.37 Predictive and Monte Carlo analysis of the probability based MMAE-based controller overlaid with the predictive analysis of the MLQG

locations. Again, this is where the probability vector points exclusively to those filters. The third plot is the simulated mean of the MMAE-based control with a probability table look-up. Except for the area around the transition from the second to third filter, the simulated performance is totally consumed by the predicted performance region.

The Monte Carlo analysis of the MMAE-based control and the M^{3}AC is presented in Figure 6.38 without the predicted region of performance overlaid. Clearly, the MMAE-based control duplicates the performance of the M^{3}AC. The blending of control that is accomplished by the M^{3}AC is duplicated by this MMAE-based control approach because it uses a gain table that has all the possible probability combinations that may

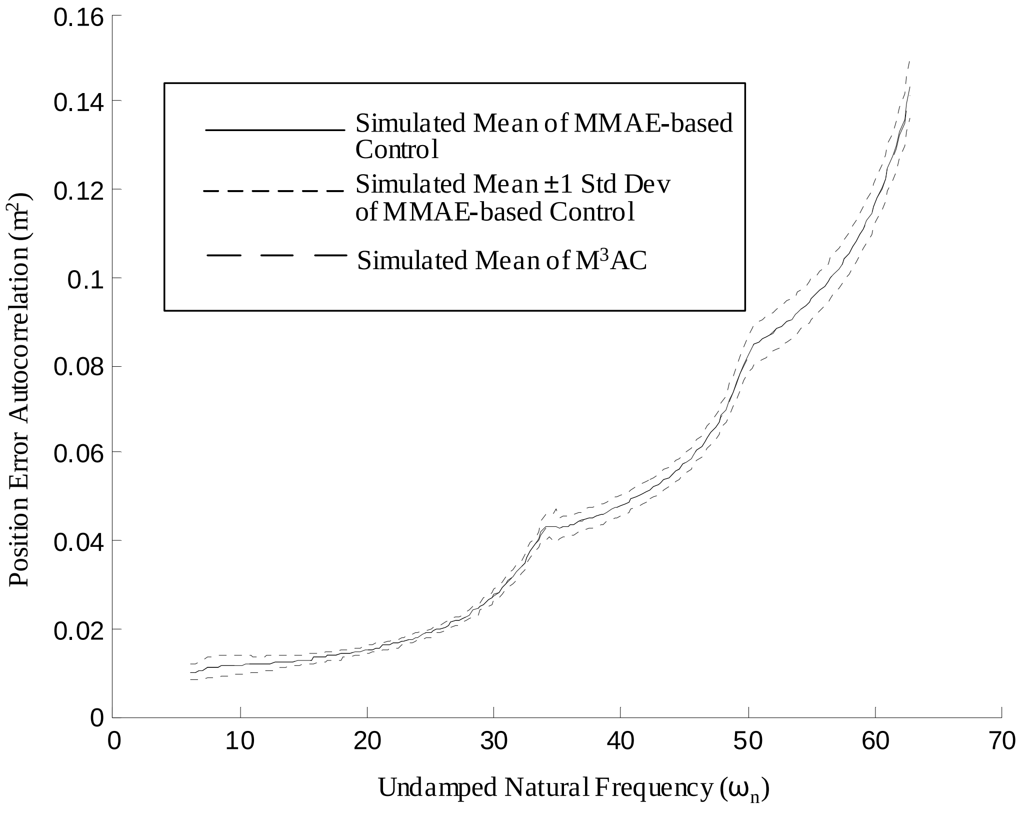

Figure 6.38 Monte Carlo analysis for the M^{3}AC and the MMAE-based control with probability based table look-up

occur during operation of the controller. Unlike the designs in the previous section in which the probability information was mapped to the assumed parameter estimate and information was lost during mapping, all probability vector information is captured in the look-up table.

This MMAE-based control structure duplicates the performance of the MMAC, but does not improve upon it. The steady state and transient response performances are identical. Since the MMAE-based control structure is more complex in implementation, it would not necessarily be the architecture of choice for most control problems. The storage requirements for the control gains could be considerable. However, the MMAE-based control does provide some online computational savings since the number of matrix multiplications is reduced.

6.6 Summary

This chapter has presented the optimal designs for the LQG controller, MMAC and MMAE-based controller developed in the previous chapters applied a two-state problem. The results show the cases for which the optimal designs for the proposed architectures had improved performance over the corresponding conventional controller architectures.

The foundation of the improved MMAC design techniques is the development of the MLQG controller. The results of implementing the MLQG controller for the example problem demonstrate its superior performance over the typical LQG controller. The generalized MLQG has the best control performance followed next by the MLQG with optimally selected controller parameter, and then the MLQG with an optimally selected filter. It is the first two design approaches that form the foundation for the modifications to the MMAC and MMAE-based control designs. Further tests of the MLQG controllers with enhanced robustness demonstrate improved performance capabilities that are not available with the typical LQG controllers.

The next set of experiments demonstrates the improvements to the MMAC by using the MLQG controller as the elemental controller. The resultant M^{3}AC significantly outperforms the control performance of the typical MMAC. The GMMAC only provided control performance enhancements similar to those of the M^{3}AC. However, a subsequent experiment demonstrates that the GMMAC can be optimized for parameter estimation while maintaining control performance superior to that of the typical MMAC.

The final set of experiments demonstrates the techniques for designing optimal MMAE-based controllers, for which several different architectures are developed. Designs for the optimal MMAE-based controllers are based on the steady state analysis that shows the MMAC and MMAE-based control have the same form. Hence, the primary revelation is that the MMAE portion must be discretized for optimal control rather than optimal parameter estimation, in order to achieve the fullest benefit for control performance.

The experiments demonstrate the differences between the controller selection schemes. Clearly, the MMAE-based control using a nominal controller may be the simplest to implement, but does not perform particularly well. The MMAE-based control with blended controller gains is architecturally the most similar to the MMAC. It also performs comparably, except at the transition points where the Monte Carlo simulation reveals poor performance. The next group of experiments tests the MMAE-based control using the parameter estimate to select the controller gain. The control performance does not quite match that of the M^{3}AC, but comes close.

A modification to the parameter lookup design is tested. Since the parameter estimate can be derived from more than one parameter combination, this approach uses the probabilities that form the parameter estimate to design the control gain. As expected, this approach is conservative since it protects against all the possible sets of probabilities that form the parameter estimate. The controller does not perform as well as one based on the simple parameter look-up approach.

The final MMAE-base controller scheme uses a probability-based table lookup approach. This requires an extensively large table sized according to the number of dimensions in the probability vector (i.e., equivalent to the number of filters). The performance of this implementation only duplicates the performance of the M^{3}AC, but does not improve upon it.

Overall, for this example problem, the blending of the individual elemental controllers that is the foundation of the MMAC provides for the best control performance. The MMAE-based control is only able to duplicate the MMAC performance and not outperform it. For the M^{3}AC, there is a slight overhead for the additional controller gain matrix multiplies, but that is all. Also, the GMMAC allows for optimal discretization for parameter estimation *and* improved control performance. It was originally postulated that the MMAE-based control was the best approach, but the GMMAC is able to provide more predictable performance.

Finally, this chapter has demonstrated all the tools that are available to the engineer for the design of optimal MLQG controllers, M^{3}AC and MMAE-based controllers, and it points to the design choices that should be considered. The example problem implementation gives a strong argument for using the MLQG controller in place of a typical LQG controller. For the multiple model adaptive control, using the M^{3}AC in place of the typical MMAC is fully warranted. In general, the M^{3}AC performs as well as the MMAE-based controller. Finally, for optimal parameter estimation with adaptive control, the GMMAC provides the best solution.

Chapter 7 - Conclusions and Recommendations

7.1 Conclusions

This research has yielded new optimal design methods for multiple model adaptive controllers (MMAC) and multiple model adaptive estimator (MMAE) based controllers as well as nonadaptive LQG controllers. The application of these design methods is valid for any cases that would be appropriate for the aforementioned controllers. The demonstrated potential for improved performance and enhanced robustness merits consideration for any possible LQG, MMAC or MMAE-based controller applications. Thus, the impact of the new discoveries is extensive.

The intent of the dissertation research has been to discover how to use the adaptive qualities of the MMAE more effectively to select a controller for applications in which the controller may be a simple gain or a full-state feedback controller. Since a portion of this research demonstrated that the MMAE-based controller can be assumed to be a generalization of the MMAC, improvements to the latter can be applied to the former. The first step that led to the most significant discovery was the development of an optimal design for the controller gains based on different design models from those used for the Kalman filters in the multiple model estimator (MME) of the MMAC. The resultant elemental controllers (i.e., Kalman filters and corresponding gains) outperformed their conventionally designed LQG counterparts. The next step was to develop optimal design for the filter and controller based on models different from the system model and potentially different from each other. These designs also outperformed their conventional counterparts. Now that the filter and controller design models are not necessarily the same as the system model, the optimal design can be accomplished for a closed set of possible system models. Hence, this area of research produced a procedure for the optimal design for enhanced robustness to possible deviations of the assumed true system.

New discoveries for optimal designs for the classical LQG control began as an adjunct area of research into multiple model adaptive control, but became central to performance improvements for both LQG and multiple model controller architectures. The anticipated results of these discoveries were addressed formally in Chapter 1 as eight hypotheses and corresponding projected contributions to multiple model adaptive control and nonadaptive LQG control. This section continues with the validation of these contributions stated in the first chapter.

The first and perhaps the most significant contribution of this research is in the aforementioned adjunct area of the modified LQG (MLQG) control design. Since the LQG controller is the most basic element of the MMAC control, it is central to further discovery. The enhancement to this basic controller element is the main contributor to the overall improvement of the MMAC architectures and has the broadest possible application to control problems. The research yielded three possible MLQG controller designs that all perform at least as well as the typical LQG controller. The first design procedure is the controller-selected MLQG controller in which the Kalman filter is designed using conventional methods (based upon a model equivalent to the system design model) and the controller design model is possibly different from the system model (in which the control design model is selected via optimization). The second method is very similar to the first except that the controller is designed using conventional LQG methods and the Kalman filter design model is possibly different from the system model. Hence, this design approach is referred to as the filter-selected MLQG controller. Finally, the generalized MLQG has the optimal design for which the filter and the controller design models are both possibly different from the system model. For the example two-state problem, the generalized MLQG outperformed the controller-selected MLQG, which in turn outperformed the filter-selected MLQG controller.

The second contribution of this research is the MLQG with enhanced robustness. This extension to the generalized MLQG yields enhanced robustness for both RMS and maximum error. The typical LQG design is only for a single system model, whereas the enhanced robustness design allows deviations of selected parameters. The deviations of the parameters actually form a closed set of system models over which the RMS or maximum error is minimized. This enhanced robustness of the LQG controller has potential application to any control problem in which there may be quantifiable and bounded deviations to system parameters.

The third contribution results from utilizing the MLQG designs as the elemental controllers for the MMAC, resulting in the modified MMAC (M^3AC). Since, for the comparison of individual controllers, the MLQG controller outperformed the conventional LQG controller, there is good basis for reasoning that the modified MMAC should outperform the conventional MMAC (based on typical LQG controllers). However, the proper discretization of the parameter space is still necessary to assure optimal performance over the range of the expected parameter values. The resultant optimal M^3AC architecture outperformed the typical MMAC for the example two-state problem. The example problem also demonstrated that the optimization of the individual

MLQG controllers will ensure that the M^{3}AC will perform at least as well as the conventional MMAC. Hence, the M^{3}AC should be considered in place of the conventional MMAC.

The fourth contribution attains a design procedure for an M^{3}AC with robustness to differences between the nominal system model and the steady-state filter model in the MME portion. This design procedure, like the previous, builds from the work with the LQG controllers and uses the MLQG with enhanced robustness as the elemental controller form. Though the implementation for the sample problem did not show significant improvements to overall performance, the design procedure may prove beneficial in other applications. Most notable might be the case for which the resultant minimization of the maximum error is significantly different from minimization of the RMS error.

The fifth contribution is a generalization of the MMAC in which the LQG controllers for the control elements are separate from the bank of Kalman filters in the MME portion. This is an extension of the work associated with the generalized MLQG. The generalized MLQG for the example two-state problem did indeed outperform the conventional LQG, but it was only slightly better than the controller-selected MLQG design. This minor improvement at the elemental controller level translated to almost identical performance between the M^{3}AC and the generalized version. However, what is significant is that the placement of the filters in the MME portion may be discretized for another performance criterion such as best parameter estimate while the full-state feedback control portion is designed for best control. The best parameter estimate could then be used for some other purpose such as performance monitoring or fault detection.

The resultant enhanced control still performs comparably to the conventional MMAC, which is only optimized for the regulation error. Finally, an additional consideration for the generalized M^{3}AC is that the filters in the MME portion may be of reduced order, compared to the order of the elemental full-state feedback controllers.

The sixth contribution of this research establishes commonality between the MMAC and MMAE-based control architectures. This is necessary in order to apply the newly discovered improvements for the MMAC to the MMAE-based controller, as well as establish the design procedures for the optimal MMAE-based controller itself. It was shown that, at steady-state conditions, the *form* of the MMAE-based control will be identical to the MMAC. However, this is not true during the transient period or before one of the filters has assumed the maximum probability. The design of the optimal controller is for steady-state conditions. For the case of lower bounding on the probability of the filters, the MMAC can only be considered a close approximation to the MMAE-based control.

The seventh contribution provides a discretization method for MMAE-based control that yields an optimal placement of the models in the MMAE with respect to a control performance criterion. Establishing the commonality between the MMAC and the MMAE-based controllers demonstrated that MMAE-based control was subject to the same trade-off between discretization for best parameter estimate and best control. The design of the optimal MMAE-based controller requires not only discretization of the MMAE, but also the design of the controller gains that the MMAE selects. This portion of the research proposed several controller schemes ranging from simply using the parameter estimate to select the controller gain to using the probability weights associated with the MMAE filters to accomplish that purpose. For each controller selection method, design algorithms for the optimal controller gains with the discretization of the MMAE were presented. The example two-state problem clearly demonstrates that the probability-based controller selection outperforms the other methods, but this technique is more complex to implement. From these results, an engineer has the tools to evaluate the trade-off between the complexity of MMAE-based controller implementations and the corresponding performances.

Finally, this research provides a modified MMAE-based control architecture that performs at least as well as the conventional MMAE-based control architecture and allows versatility in the control scheduling for possible values of uncertain parameters of the system as determined by parameter estimates. This allows the engineer to discretize the filters in MMAE portion for a criterion other than optimal control and still use the parameter estimate (or probability associated with the estimate) for selecting the control. The most likely choice of performance criterion for the MMAE would be optimal parameter estimation. An optimal parameter estimate then could be used for purposes other than control, such as monitoring operating conditions to detect performance degradation or failures. The performance enhancement of this architecture comes from the benefits of the MLQG controllers over the conventional LQG controllers. However, it must be emphasized that the modified MMAE-based control architecture in which the MMAE *and* controller gains are optimized for the control criterion provides the best performance.

7.2 Recommendations

This research has yielded several key discoveries that routinely should be applied to typical LQG control problems as well as to those involving multiple model adaptive control. However, there are additional areas of refinement that might provide ease of implementation and additional framework for applications.

The approach for selecting the design models in the implementation experiments in Chapter 6 was very basic. In general, the MATLAB *fminsearch* minimization routine [31] was used to find the optimal value of the parameter of the specified design model. When both the filter and controller models were different from the system model, an application of the minimization had to occur within a minimization. For the example implementation of Chapter 6, many of the minimizations settled to local minima. This was especially problematic for the M^3AC and modified MMAE-based controllers. The example problem was only a two-state system with one uncertain design parameter. In order to make the design methods developed in this research more readily usable, a better approach to selecting the filter, controller or both within the framework of the MLQG would be beneficial.

As stated, the example problem was a two-state system with one uncertain parameter. The position error performance curves were only second order with what turned out to be large ranges of flat response before going asymptotically to the maximum. This effect potentially limited the effectiveness of the robustness techniques. The average maximum error over any given intervals was not very different. To demonstrate the benefits of the enhanced robustness for the MLQ further, M^3AC and modified MMAE-based controllers, higher order example problems should be investigated.

Another potentially productive area of additional research is reduced order models between those in the MME and those for the elemental controllers for the generalized MMAC and the MMAE-based control approaches. A significantly reduced order model for the filters used to compute the probability weighting may be adequate to select the higher order controller. This has the potential to reduce the complexity of implementation for large problems.

Of these three additional areas for further research, better methods for determining the filter and controller models would have the greatest impact. The MLQG and M^3AC should have truly widespread application. However, they will only gain acceptance with successful application of the techniques.

Bibliography

1. Athans, M., et al. “The Stochastic Control of the F-8C Aircraft Using a Multiple Model Adaptive Control (MMAC) Method-Part1: Equilibrium Flight,” *IEEE Transactions on Automatic Control*, Vol. AC-22, No. 5, pp. 768-780, October 1977.

2. Athans, M.,“The Discrete Time Linear-Quadratic–Gaussian Stochastic Control Problem,” *Annals of Economic and Social Measurement*. Vol. 1, No. 4. 1972

3. Baram, Yoram, *Information, Consistent Estimation and Dynamic System Identification*. PhD Thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, November 1976.

4. Baram, Yoram and Nils R. Sandell, Jr. “An Information Theoretic Approach to Dynamical Systems Modeling and Identification,” *IEEE Transactions on Automatic Control*, Vol. AC-23 No. 1, pp. 61-66, February 1978.

5. Baram, Yoram and Nils R. Sandell, Jr. “Consistent Estimation on Finite Parameter Sets with Application to Linear Systems Identification,” *IEEE Transactions on Automatic Control*, Vol. AC-23 No. 3, pp. 451-453, June 1978.

6. Bernstein, Dennis and Wassim Haddad. “LQG Control with an H_∞ Performance Bound: A Riccati Equation Approach,” *IEEE Transactions on Automatic Control*, Vol. 34, No. 3, pp. 293-305, March 1989.

7. Blom, Henk A. P. and Yaakov Bar-Shalom. “The Interacting Multiple Model Algorithm for Systems with Markovian Switching Coefficients,” *IEEE Transaction on Automatic Control*, Vol. 33, No. 8, pp. 780-783, August 1988.

8. Cliz, Mekal and K. S. Narendra. “Adaptive Control of Robotic Manipulators using Multiple Modes and Switching,” *International Journal of Robotics Research*, Vol. 15, No. 6, pp. 592-610, December 1996.

9. Cliz, Mekal and K. S. Narendra. “Multiple Model Based Adaptive Control of Robotic Manipulators,” *Proceedings of the 33rd Conference on Decision and Control*, Orlando, Florida, pp. 1305-1310, December 1994.

10. Dubowsky, S. and D. T. DesForges. “The Application of Model Reference Adaptive Control to Robotic Manipulators,” *Journal of Dynamic Systems, Measurement, and Control*, Vol. 101, pp. 193-200, September, 1979.

11. Eide, Peter K. *Implementation and Demonstration of a Multiple Model Adaptive Estimator Failure Detection System for the F-16*. MS Thesis, AFIT/GE/ENG/94D-06, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1994.

12. Eide, Peter K. and Peter S. Maybeck. "An MMAE Failure Detection System for the F-16," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 32, No. 3, July, 1996.

13. Fisher, K. A. and Peter S. Maybeck. "Multiple Model Adaptive Estimation with Filter Spawning," *Proceedings of the 2000 American Control Conference,* Chicago, Illinois, pp. 2326-2331, June 2000.

14. Goodwin, Graham and David Mayne. "Continuous-Time Stochastic Model Reference Adaptive Control," *IEEE Transactions on Automatic Control*, Vol. 36, No. 11, pp. 1254- 1263, November, 1991.

15. Greene, C. S. and A. S. Willsky. "An Analysis of the Multiple Model Adaptive Control Algorithm," *Proceedings of the19th IEEE Conference on Decision and Control*, pp. 1142-1145, Albuquerque, New Mexico, 1980.

16. Griffin, Gordon C. *Control of a Large Space Structure Using Multiple Model Adaptive Estimation and Control Techniques*, MS Thesis, AFIT/GE/ENG/94D-14, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1994.

17. Griffin, Gordon C. and Peter S. Maybeck. "MMAE/MMAC Techniques Applied to Large Space Structures Bending with Multiple Uncertain Parameters," *Proceedings of the 34th Conference on Decision and Control*, pp. 1153-1158, New Orleans, Louisiana, December 1995.

18. Griffin, Gordon C. and Peter S. Maybeck. "MMAE/MMAC Control for Bending with Multiple Uncertain Parameters," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 33, No. 3, pp. 903-912, July 1997.

19. Gustafson, John A. *Control of a Large Flexible Space Structure Using Multiple Model Adaptive Algorithms*. MS Thesis, AFIT/GE/ENG/91D-22, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1991.

20. Gustafson, John A. and P. S. Maybeck. "Flexible Space Structure Control Via Moving-Bank Multiple Model Algorithms, " *IEEE Transaction on Aerospace and Electronic Systems*, Vol. 30, No. 3, pp. 750-757, July 1994.

21. Hanlon, Peter D. *Practical Implementation of Multiple Model Adaptive Estimation Using Neyman-Pearson Based Hypothesis Testing and Spectral Estimation Tools*. Ph.D. Dissertation, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, September 1996.

22. Hentz, K. P., Feasibility *Analysis of Moving Bank Multiple Model Adaptive Estimation and Control Algorithms*. Masters Thesis, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1984.

23. Kwakernaak, H. and Sivan, R., *Linear Optimal Control Systems*. New York: Wiley (Interscience), 1972.

24. Li, Xiao-Rong and Yaakov Bar-Shalom. "Multiple-Model Estimation with Variable Structure," *IEEE Transactions on Automatic Control*, Vol 41, No. 4, pp. 479-493, April, 1996.

25. Leahy, M. B., et. al., "Experimental Evaluation of A Puma Manipulator Model-Referenced Adaptive Controller," *Proceedings of the 26th IEEE Conference on Decision and Control*, pp. 2196-2201, Los Angeles, California, December 1987.

26. Lund, Eivind J. *On-line Discrimination and Estimation in Multiple Regime Systems*. Ph.D. Dissertation, Division of Engineering Cybernetics, The Norwegian Institute of Technology University of Trondheim, June 1992.

27. Lund, Eivind J., et al. "Multiple Model Estimation with Inter-Residual Distance Feedback," *Modeling, Identification and Control*, Vol. 13, No. 3, pp. 127-140, 1992.

28. Maciejowski, J. M. *Multivariable Feedback Design*, Addison-Wesley Publishing Company, New York, 1989.

29. Magill, D. T. "Optimal Adaptive Estimation of Sampled Stochastic Processes," *IEEE Transactions on Automatic Control,* AC-10, No. 5, 434-439, 1965.

30. Makila, Pertti M. "Multiple Models, Multiplicative Noise and Linear Quadratic Control-Aogorithmic Aspects," *International Journal of Control*, Vol 54, No. 4, 921-941, 1991.

31. Mathworks, *MATLAB 6.5 Student Version*, The Mathworks Inc., 2002.

32. Matthes, James R. *Model Selection for the Multiple Model Adpative Algorithm for In-Flight Simulation.* MS Thesis, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, March 1985.

33. Maybeck, Peter S. *Combined Estimation of States and Parameters for On-Line Applications.* Rep. T-557, Ph.D. Dissertation, MIT, Cambridge Massachusetts, February, 1972.

34. Maybeck, Peter S. *Stochastic Models, Estimation and Control I.* New York: Academic Press Inc., 1979. Republished, Arlington Virginia, Navtech, 1994.

35. Maybeck, Peter S. *Stochastic Models, Estimation and Control II.* New York: Academic Press Inc., 1982. Republished, Arlington Virginia, Navtech, 1994.

36. Maybeck, Peter S. *Stochastic Models, Estimation and Control III.* New York: Academic Press Inc., 1982. Republished, Arlington Virginia, Navtech, 2001

37. Maybeck, Peter S. "Moving-Bank Multiple Model Adaptive Estimation and Control Algorithms: An Evaluation," *Control and Dynamic Systems*, Vol. 31, 1989.

38. Maybeck, Peter S. and Peter D. Hanlon. "Performance Enhancement of a Multiple Model Adaptive Estimator," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 27, No. 3, 470-479, May 1991.

39. Maybeck, Peter S. and Richard D. Stevens. "Reconfiguarable Flight Control Via Multiple Model Adaptive Control Methods," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 27, No. 3, 470-479, May 1991.

40. Maybeck, Peter S. Discussion with Thomas Brehm, Aug 1998.

41. Menke, Timothy E. *Multiple Model Adaptive Estimation Applied to the VISTA F-16 with Actuator and Sensor Failures*. MS Thesis, AFIT/GE/ENG/92j-01, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, June 1992.

42. Menke, Timothy E. and Peter S. Maybeck. "Sensor/Actuator Failure Detection in the VISTA F-16 by Multiple Model Adaptive Estimation," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 31, No.4, 1218-1228, October 1995.

43. Meyn, Sean and Lyndon Browm. "Model Reference Adaptive Control of Time Varying and Stochastic Systems," *IEEE Transactions on Automatic Control*, Vol. 38, No. 12, 1738-1753, December 1993.

44. Miller, Mikel M. *Modified Multiple Model Adaptive Estimation (M^3AE) For Simultaneous Parameter and State Estimation*, PhD Dissertation, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, March 1998..

45. Muravez, Randall J. *Multiple Model Adaptive Estimation and Predication with the Harmonically Balanced Kalman Filter Bank.* MS Thesis, California Polytechnic University, Pomona, California, December 1989.

46. Narendra, K. S., et al. "Adaptation and Learning Using Multiple Models, Switching and Tuning," *IEEE Control Systems*, pp. 37-51, June 1995.

47. Narendra, K. S. and Jeyendran Balakrishnan. "Improving Transient Response of Adaptive Control Systems Using Multiple Models and Switching," *IEEE Transactions on Automatic Control*, Vol. 39, No. 9, pp. 1861-1866, September 1994.

48. Narendra, K. S. and Jeyendran Balakrishnan. "Adaptive Control Using Multiple Models," *IEEE Transactions on Automatic Control*, Vol. 42, No. 2, pp. 171-187, February, 1997.

49. Rangan, S. and K. Poolla. "Asymptotic Performance in Adaptive H_∞ Control," *Proceedings of the 35th IEEE Conference on Decision and Control*, pp. 3755-3759, Kobe, Japan, December 1996.

50. Rangan, S. and K. Poolla. "Multimodel Adaptive Adaptive H_∞ Control," *Proceedings of the 35th IEEE Conference on Decision and Control,* Kobe, Japan, pp. 1928-1933, Dec 1996.

51. Ren, Wi and P. Kumar. "Stochastic Adaptive Prediction and Model Reference Control," *IEEE Transactions on Automatic Control*, Vol. 39, No. 10, pp. 2047-2059, October 1994.

52. Schiller, Gregory J. *Control of a Large Space Structure Using Multiple Model Adaptive Estimation and Control Techniques*. MS Thesis, AFIT/GE/ENG/93D-02, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1993.

53. Schiller, Gregory J. and Peter S. Maybeck. "Space Structure Control Using Multiple Model Adaptive Estimation and Control," *Proceedings of the 13th IFAC Symposium on Automatic Control in Aerospace – Aerospace Control '94*, Palo Alto, California, pp. 216-221, September 1994.

54. Schiller, Gregory J. and Peter S. Maybeck. "Control of a Large Space Structure Using MMAE/MMAC Techniques," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 33, No. 4, pp. 1122-1131, October 1997.

55. Seraji, H. " A New Approach to Adaptive Control of Manipulators," *Journal of Dynamic Systems, Measurement and Control*, Vol. 109, pp. 193-202, September, 1987.

56. Sheldon, Stuart N. *An Optimal Parameter Discretization Strategy for Multiple Model Adaptive Estimation and Control*. Ph.D. Dissertation, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1989.

57. Sheldon, Stuart N. and Peter S. Maybeck. "An Optimizing Design Strategy for Multiple Model Adaptive Estimation and Control," *IEEE Transactions on Automatic Control*, Vol. 38, No. 4, pp. 651- 654, April 1993.

58. Stepaniak, Michael J. *Multiple Model Adaptive Control of the VISTA F-16*. MS Thesis, AFIT/GE/ENG/95D-26, School of Engineering, Air Force Institue of Technology, Wright-Patterson AFB, OH, December 1995.

59. Stepaniak, Michael and Peter S. Maybeck. "MMAE-Based Control Redistribution Applied to the VISTA F-16," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 34, No. 4, pp. 1249-1260, October 1998.

60. Stevens, Richard D. *Characterization of a Reconfigurable Multiple Model Adaptive Controller Using a STOL F-15 Model*. MS Thesis, AFIT/GE/ENG/89D-52, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1989.

61. Tao, Gang. "Robustness of MRAC Schemes," *Proceedings of the American Control Conference*, Albuquerque, New Mexico, pp. 744-745, June 1997.

62. Tellman, Larry D. *Multiple Model-Based Robot Control: Development and Initial Evaluation*. MS Thesis, AFIT/GE/ENG/88D-55, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1988.

63. Vasquez, Juan R. *New Algorithms for Moving-Bank Multiple Model Adaptive Estimation*, PhD Dissertation, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, June 1998.

64. Vasquez, Juan R. and P. S. Maybeck. "Density Algorithm Based Moving-Bank MMAE," *Proceedings of the 38th Conference on Decision and Control*, Phoenix, Arizona, pp. 4117-4122, December 1999.

65. Vasquez, Juan R. and P. S. Maybeck. "Enhanced Motion and Sizing of Bank in Moving-Bank MMAE," *Proceedings of the 1999 American Control Conference*, San Diego, California, pp. 1555-1562, June 1999.

VITA

Thomas E. Brehm graduated from Bishop Watterson High School Columbus, Ohio. He entered the undergraduate program at the Ohio State University where he graduated with a Bachelor of Science degree in Electrical Engineering in December 1986. Following graduation be began civil service with the United States Air Force at the Foreign Technology Division (FTD) at Wright Patterson AFB, OH. While working full time at FTD, he entered the graduate program at Wright State University in Dayton, where he graduated with a Masters of Science in Computer Engineering in April 1994. Thomas began the graduate program at the Air Force Institute of Technology on a part time basis in September, 1995.

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)* 23-03-2004	**2. REPORT TYPE** Doctral Dissertation	**3. DATES COVERED** *(From - To)* Nov 1998- Mar 2004

4. TITLE AND SUBTITLE OPTIMAL DESIGN OF GENERALIZED MULTIPLE MODEL ADAPTIVE CONTROLLERS	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Brehm, Thomas E.	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Intstitutue of Technology (AFIT/ENG) Graduate School of Engineering and Management 2950 Hobson Way, Building 640 WPAFP,OH 45433-7765	**8. PERFORMING ORGANIZATION REPORT NUMBER** AFIT/DS/ENG/04-01
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFRL/VACA (AFMC) Attn: Dr. Andrew G. Sparks 2210 Eighth Street, Building 146 Wright Patterson Air Force Base, OH 45433-7531 (937) 255-8682 DSN 785-8682	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Advanced analysis and optimal design techniques that achieve performance improvement for multiple model adaptive control (MMAC) and multiple model adaptive estimation (MMAE) based control are developed and tested. An adjunct area of research yielded modified linear quadratic Gaussian (LQG) control design techniques that also can be applied to nonadaptive control. For the Modified LQG (MLQG) controller, the proposed designs remove the assumption that the Kalman filter and the controller gain matrix design are necessarily based on the same model as the best system model. The proposed modified MMAC (M3AC) architectures use the MLQG controller. The performance improvements for the MLQG controller carry over to the M3AC architectures as well as to the MMAE-based control architecture. Design approaches developed for the M3AC are applied to the MMAE-based control with similar performance improvements. Analyses of a representative example of the new design implementations demonstrate the performance improvements of the proposed architectures by comparing the results with those of the typical MMAC and LQG implementations. Though incidental to this research, the performance enhancement of the MLQG controller itself has proven to be significant and the possibilities for application to non-adaptive control transcend this research.

15. SUBJECT TERMS

ADAPTIVE CONTROL SYSTEMS, REGULATORS, KALMAN FILTERING, MODELS

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Peter S. Maybeck, PhD, DAF (AFIT/ENG)
U	U	U	UU	298	**19b. TELEPHONE NUMBER** *(Include area code)* (937) 255-3636, ext 4581;peter.maybeck@afit.edu